Inference of Low-Dimensional Latent Structure in High-Dimensional Data

by

Haojun Chen

Department of Electrical and Computer Engineering
Duke University

Date: _____

Approved:

_____
Lawrence Carin, Supervisor

_____
Rebecca Willett

_____
Matthew Reynolds

_____
David Dunson

_____
Sayan Mukherjee

Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Electrical and Computer Engineering in the Graduate School of Duke University
2011

Abstract
(EE)

Inference of Low-Dimensional Latent Structure in High-Dimensional Data

by

Haojun Chen

Department of Electrical and Computer Engineering
Duke University

Date: _____

Approved:

_____
Lawrence Carin, Supervisor

_____
Rebecca Willett

_____
Matthew Reynolds

_____
David Dunson

_____
Sayan Mukherjee

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Electrical and Computer Engineering
in the Graduate School of Duke University
2011

Abstract

The problem of learning a latent model for sparse or low-dimensional representation of high-dimensional data has attracted significant attention for many years. This thesis focuses on learning latent models for sparse or low-dimensional representation of images, dynamic data, and documents with Bayesian nonparametrics. The thesis consists of three parts.

First, nonparametric Bayesian methods are considered for recovery of imagery based upon compressive measurements. A truncated beta-Bernoulli process is employed to infer an appropriate dictionary for the test data, and also for image recovery. In the context of compressive sensing, significant improvements in image recovery are manifested using learned dictionaries, relative to using standard orthonormal image expansions. The compressive-measurement projections are also optimized for the learned dictionary. Spatial inter-relationships within imagery are exploited through use of the Dirichlet and probit stick-breaking processes. Several example results are presented, with comparisons to other state-of-the-art methods in the literature.

Second, hierarchical Bayesian methods are employed to learn a reversible statistical embedding. The proposed embedding procedure is connected to spectral embedding methods (*e.g.*, diffusion maps and Isomap), yielding a new *statistical* spectral framework. The proposed approach allows one to discard the training data when embedding new data, allows synthesis of high-dimensional data from the embedding space, and provides accurate estimation of the latent-space dimensionality. Hier-

archical Bayesian methods are also developed to learn a nonlinear dynamic model in the low-dimensional embedding space, allowing joint analysis of multiple types of dynamic data, sharing strength and inferring inter-relationships. In addition to *analyzing* dynamic data, the learned model also yields effective synthesis. Example results are presented for statistical embedding, latent-space dimensionality estimation, and analysis and synthesis of high-dimensional (dynamic) motion-capture data.

Third, a new hierarchical tree-based topic model is developed, based on nonparametric Bayesian techniques. The model has two unique attributes: (i) a child node in the tree may have more than one parent, with the goal of eliminating redundant sub-topics deep in the tree; and (ii) parsimonious sub-topics are manifested, by removing redundant usage of words at multiple scales. The depth and width of the tree are unbounded within the prior, with a retrospective sampler employed to adaptively infer the appropriate tree size based upon the corpus under study. Excellent quantitative results are manifested on five standard data sets, and the inferred tree structure is also found to be highly interpretable.

Contents

List of Tables

List of Figures

List of Abbreviations and Symbols

BP	beta process
CS	compressive sensing
DP	Dirichlet process
FA	factor analysis
GP-LVM	Gaussian process linear variable models
HDP	hierarchical Dirichlet process
HMT	hierarchical Markov tree
i.i.d.	independently and identically distributed
LDA	latent Dirichlet allocation
MAP	maximum a posteriori
MCMC	Markov chain Monte Carlo
MFA	mixtures of factor analyzers
nCRP	nested Chinese restaurant process
NSR	nonlinear spectral regression
RBF	radial basis function
PCA	principal components analysis
PSBP	probit stick-breaking process
RVM	relevance vector machine
RMSE	root mean squared error
VB	variational Bayesian

Acknowledgements

First of all, I would like to express my heartfelt gratitude to my advisor, Dr. Lawrence Carin, for his guidance, inspiration, encouragement and patience throughout my graduate studies. Without his tremendous support, this work would not be finished. His influence on my thought and working attitude will continue beyond the completion of this thesis.

I would like to thank Dr. David Dunson, for his insightful suggestions and valuable comments on my research. I also want to thank the other committee members: Dr. Matt Reynolds, Dr. Rebecca Willett, and Dr. Sayan Mukherjee, for their time and comments on this thesis.

Special thanks to Dr. Jorge Silva for his great help. I would also like to thank all other my research group members, including Dr. Xuejun Liao, Dr. Qi An, Dr. Bo Chen, Dr. Lihan He, Dr. Dehong Liu, Dr. John Paisley, Dr. Iulian Pruteanu, Dr. Yuting Qi, Dr. Chunping Wang, Minhua Chen, Lingbo Li, Eric Wang, Zhengming Xing and Mingyuan Zhou, for their collaboration, assistance and valuable discussions. I really enjoy the time spent with them and benefited a lot from such an experience.

Last and foremost, I especially thank my parents for their lifelong encouragement and support. Their love means a lot to me and will accompany me forever. Finally, I am very thankful to my wife, whose loving support has been and always will be my most precious possession on earth, and my daughter, who is a source of endless love and joy.

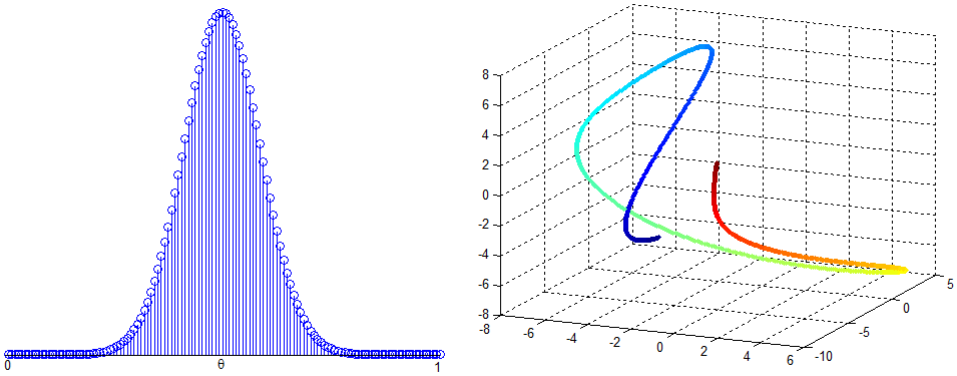
1

Introduction

The growth of information technology has greatly improved the quality of people's lives. From wired network to wireless network, DVD to Blu-ray, digital camera to camcorder, HDTV to 3D-TV, and mobile phone to NirvanaPhone, people can freely and immediately access and/or transfer vocal, pictorial and textual information. Meanwile, the quantity of raw data demanded from the sensing systems has dramatically increased. Consequently, the dimension or size D of the typical desired signal of these systems keeps increasing. This undoubtedly causes many difficulties for the acquisition, processing, storage, transmission and analysis of these high-dimensional data.

Fortunately, in many cases, the high-dimensional data may be represented by a latent sparse or low-dimensional model, with only $d \ll D$ degrees of freedom. Inferring such a model often reduces in the required quantity of high-dimensional data needed for the sensing system. As a simple example (taken from [Wak10]), we consider a set of 128-dimension signals generated by shifting the Gaussian pulse $f(n-\theta)$ in Figure 1.1(a). Although the number of samples of the signal is large, the only degree of freedom is the scalar shift parameter θ. If we project this set of

signals onto a random three-dimensional subspace, then the signals will, with high probability, form a twisting curve that does not self-intersect (Figure 1.1(b)). Thus, very few measurements are needed to represent the characteristics of such a signal. In the real world, many high-dimensional data actually obey some kind of latent sparse or low-dimensional model, such as images of human faces and handwritten digits [TP91, HDR97, TdSL00]. Further, the low-dimensional latent parameters may often be linked to underlying physics or biology, of importance for interpretation. For example, the d dimensional parameter θ can be view as the uncertainty of the 1-D arrival time of the signal(as in Figure1.1(a)), the 2-D fingers extension and wrist rotation of the hand images [TdSL00] and the 3-D up-down pose, left-right pose and lighting direction of the face images [TdSL00].



(a) (b)

FIGURE 1.1: (a) example from a shifted Gaussian, with peak shift θ. (b) projections of *multiple* such signals onto a random 3D subspace, with the different points on the curve corresponding to different shifts θ.

However, two problems still need to be solved: (i) how to learn such a latent sparse or low-dimensional model and (ii) how to choose dimension d. Motivated by these two challenges, we are interested in learning a category of latent sparse or low-dimensional models for representation of high-dimensional data with nonparametric probabilistic methods. Although this thesis focuses on images, dynamic data, and

text data here, the proposed models can be extended to many other applications.

The rest of this chapter is organized as follows: nonparametric Bayesian prior and factor models are introduced in Section 1.1 and 1.2, respectively. The organization of this thesis is provided in Section 1.3.

1.1 Nonparametric Bayesian Priors

1.1.1 Dirichlet Process (DP)

A Dirichlet Process [Fer73], $DP(\alpha, G_0)$, is a measure on measures where α is a positive concentration parameter and G_0 is a probability measure on some measurable space. A draw from DP is defined as $G \sim DP(\alpha, G_0)$, where G is also a measure and $E[G] = G_0$.

The stick-breaking construction [Set94] for G can be explicitly represented by

$$G = \sum_{k=1}^{\infty} w_k \delta_{\boldsymbol{\theta}_k^*} \tag{1.1}$$

where the infinite random variable $\boldsymbol{\theta}_k^*$ are independently sampled from the base measure G_0, $\delta_{\boldsymbol{\theta}_k^*}$ is an atom at $\boldsymbol{\theta}_k^*$ and w_k is the weight for the sample of each G that choose the atom $\boldsymbol{\theta}_k^*$, with $0 \leq w_k \leq 1$ and $\sum_{k=1}^{\infty} w_k = 1$. The random weight variables $\{w_k\}_{k=1,2,\cdots,\infty}$ are dependent on scale parameter α to break a unit-length "stick":

$$w_k = V_k \prod_{i=1}^{k-1} (1 - V_i) \tag{1.2}$$

where $V_k|\alpha \overset{iid}{\sim} Beta(1, \alpha)$ and $\boldsymbol{\theta}_k^*|G_0 \overset{iid}{\sim} G_0$. Note that α controls the similarity between G and G_0. If α is large enough, each V_k drawn from $Beta(1, \alpha)$ will be relatively small. This implies that we will have many very short sticks. Consequently, the weights w_k for an infinite number of $\boldsymbol{\theta}_k^*$ are all very small and therefore G will approach the base measure G_0. On the other hand, if α is small, each π_k drawn

from $Beta(1, \alpha)$ will be relatively large, which leads to a few large sticks with the remaining sticks very small. As a result, G will only have a large mass on a small subset $\{\boldsymbol{\theta}_k^*\}_{k=1,2,\cdots,\infty}$, (those $\boldsymbol{\theta}_k^*$ corresponding to the large sticks w_k). In practice, a Gamma prior is usually placed on α.

1.1.2 Beta Process (BP)

The two-parameter beta process (BP) was developed in [PC09b, TJ07], to which the reader is referred for further details; we here only provide those details of relevance for this thesis. The BP with parameters $a > 0$ and $b > 0$, and base measure H_0, is represented as $\text{BP}(a, b, H_0)$, and a draw $H \sim \text{BP}(a, b, H_0)$ may be represented as

$$\begin{aligned} H(\boldsymbol{\psi}) &= \sum_{k=1}^{K} \pi_k \delta_{\boldsymbol{\psi}_k} \\ \pi_k &\sim \text{Beta}(a/K, b(K-1)/K) \\ \boldsymbol{\psi}_k &\sim H_0 \end{aligned} \tag{1.3}$$

with this a valid measure as $K \to \infty$. Therefore, $H(\boldsymbol{\psi})$ represents a vector of K probabilities, with each associated with a respective atom $\boldsymbol{\psi}_k$. In the limit $K \to \infty$, $H(\boldsymbol{\psi})$ corresponds to an infinite-dimensional vector of probabilities, and each probability has an associated atom $\boldsymbol{\psi}_k$ drawn i.i.d. from H_0.

Using $H(\boldsymbol{\psi})$, we may now draw N *binary* vectors, the ith of which is denoted $\boldsymbol{z}_i \in \{0, 1\}^K$, and the kth component of $\boldsymbol{z}_i$ is drawn $z_{ik} \sim \text{Bernoulli}(\pi_k)$. These N binary column vectors are used to constitute a matrix $\mathbf{Z} \in \{0, 1\}^{K \times N}$, with ith column corresponding to $\boldsymbol{z}_i$; the kth row of $\mathbf{Z}$ is associated with atom $\boldsymbol{\psi}_k$, drawn as discussed above. Note that the choice of a and b will control the sparseness of $\mathbf{Z}$. If the probability vector $\boldsymbol{\pi}_j$ is marginalized out, the number of non-zero elements of $\boldsymbol{z}_i$ is drawn from $\text{Binomial}(K, a/(a + b(K-1)))$, and the expected number of ones is $aK/(a + b(K-1))$. As $K \to \infty$ the number of nonzero components is distributed

Poisson(a/b) and the expected number of ones is a/b.

1.2 Factor Models

1.2.1 Factor analysis (FA)

Factor analysis is a classical approach for modeling high-dimensional data by low-dimensional latent variables (or factors), widely used in chemistry, social science, statistics and machine learning field. In factor analysis, the high-dimensional data in $\mathbb{R}^D$ are described as a linear combination of a potentially small number of factor loadings, which are also in $\mathbb{R}^D$.

Given a set of data $\{\boldsymbol{y}_i\}_{i=1,N}$, where $\boldsymbol{y}_i \in \Re^D$, and D is large. The matrix $\boldsymbol{Y} \in \Re^{D\times N}$ has columns defined by the vectors in the set $\{\boldsymbol{y}_i\}_{i=1,N}$. A factor-analysis (FA) model may be represented

$$\boldsymbol{y}_i = \boldsymbol{A}\boldsymbol{x}_i + \boldsymbol{\mu} + \boldsymbol{\epsilon}_i \tag{1.4}$$

where the kth column of factor loading matrix $\boldsymbol{A}$ is drawn $\boldsymbol{a}_k \sim \mathcal{N}(\boldsymbol{0}, \beta_k^{-1}\boldsymbol{I}_D)$, sample dependent factor score $\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_d)$, $\boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{0}, \gamma^{-1}\boldsymbol{I}_D)$, $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \boldsymbol{\Psi})$ and $\boldsymbol{\Psi}$ is a diagonal matrix. Therefore, the conditional density of $\boldsymbol{y}_i$ is

$$p(\boldsymbol{y}_i|\boldsymbol{A}, \boldsymbol{\mu}, \boldsymbol{\Psi}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{A}\boldsymbol{A}^T + \boldsymbol{\Psi}) \tag{1.5}$$

Each $\boldsymbol{y}_i$ is consequently assumed drawn from a *linear* subspace defined by the columns of $\boldsymbol{A}$, and typically $\boldsymbol{A}\boldsymbol{A}^T$ is of low rank (*i.e.*, $\boldsymbol{y}_i$ is drawn from a low-dimensional portion of $\Re^D$ centered about $\boldsymbol{\mu}$). Factor analysis is closely related to probabilistic principal component analysis (PCA) [TB99b]. These two methods become mathematically equivalent if the diagonal elements in the noise covariance matrix $\boldsymbol{\Psi}$ have the same value in (1.4).

1.2.2 Mixture of factor analyzers (MFA)

A natural extension to a nonlinear model is achieved by assuming $\boldsymbol{y}_i$ is drawn from a *mixture* of factor analyzers models (MFA) [GB00, TB99a] and the number of mixture components may be inferred nonparametrically using the Dirichlet process (DP) [Fer73]. The nonparametric MFA [CSP+10] can be written as

$$\boldsymbol{y}_i = \boldsymbol{A}_{z(i)}\boldsymbol{x}_i + \boldsymbol{\mu}_{z(i)} + \boldsymbol{\epsilon}_i \tag{1.6}$$

with $\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{0}, \beta_{z(i)}^{-1}\boldsymbol{I}_d), z(i) \sim \sum_{j=1}^{\infty} w_j\delta_j$, $w_j = V_j\prod_{l=1}^{j-1}(1-V_l)$; $V_j \sim \text{Beta}(1,\alpha)$, with $\{\boldsymbol{A}_j\}_{j=1,\infty}$ and $\{\boldsymbol{\mu}\}_{j=1,\infty}$ drawn as above (in practice the number of "sticks" π_j is truncated [IJ01], and when truncating to J sticks, $V_J = 1$). Gamma hyperpriors are placed on DP parameter α and $\{\beta_j\}_{j=1,J}$.

The DP allows inference of an appropriate number of FA mixture components. It is also of interest to nonparametrically learn the latent-space dimensionality d. To construct each $\boldsymbol{A}_j$, we employ a spike-slab prior [Wes03]. Toward this end, we augment the above model, and assume $\boldsymbol{A}_j \in \Re^{N\times K}$ and $\boldsymbol{x}_i \in \Re^K$, where K is a large integer (large relative to the anticipated d, with d inferred by the number of columns in $\boldsymbol{A}_j$ that are ultimately used in forming $\boldsymbol{A}_{z(i)}\boldsymbol{x}_i$).

Assume the stick-breaking representation of the aforementioned DP is truncated to J components: $z(i) \sim \sum_{j=1}^{J} w_j\delta_j$, with $\sum_{j=1}^{J} w_j = 1$. For each of the J mixture components we draw K-dimensional probability vectors

$$\boldsymbol{\pi}_j \sim \prod_{k=1}^{K} \text{Beta}(a/K, b(K-1)/K) \tag{1.7}$$

where $a > 0$ and $b > 0$. For each of the K columns of $\boldsymbol{A}_j$ we draw a K-dimensional binary vector $\boldsymbol{b}_j \sim \prod_{k=1}^{K} \text{Bernoulli}(\pi_{jk})$ where π_{jk} is the kth component of $\boldsymbol{\pi}_j$. Finally, each component of the kth column of $\boldsymbol{A}_j$ is multiplied by b_{jk}, thereby constituting a factor loading matrix $\boldsymbol{A}_j$ that typically has some columns that are not used in the data construction (allowing us to infer the dimensionality of the latent space).

The nonparametric MFA model [CSP+10] may now be summarized as

$$\boldsymbol{y}_i = \boldsymbol{A}_{z(i)}\boldsymbol{\Lambda}_{z(i)}\boldsymbol{x}_i + \boldsymbol{\mu}_{z(i)} + \boldsymbol{\epsilon}_i \tag{1.8}$$

with $\boldsymbol{\Lambda}_{z(i)} = \text{diag}(b_{z(i),1}, \ldots, b_{z(i),K})$, with $\boldsymbol{b}_j$ and $z(i) \sim \sum_{j=1}^{J} w_j \delta_j$ as discussed above.

1.2.3 Latent Dirichlet allocation (LDA)

Latent Dirichlet Allocation (LDA) [BNJ03], also known as multinomial PCA (MPCA), is a generative model, widely used to infer low-dimensional latent semantic information (topics) associated with a corpus of documents, where a topic is a distribution over words. For each document, LDA first generate a distribution over topics, and then generate each word from a topic drawn from this distribution. Suppose we have V terms in the dictionary, T topics, D documents and N_d words in document d. The generative process of LDA can be summarized as follows:

1. For each topic $t \in 1, \ldots, T$, draw topic distributions $\boldsymbol{\phi}_t \sim \text{Dirichlet}(\gamma)$
2. For document d
   (a) Draw distribution over topics $\boldsymbol{\theta}_d \sim \text{Dirichlet}(\alpha)$
   (b) For the i-th word:
      i. Draw topic indicator $z_{di} \sim \text{Mult}(\boldsymbol{\theta}_d)$
      ii. Draw word $w_{di}|z_{di} \sim \text{Mult}(\boldsymbol{\phi}_{z_{di}})$

According to the above generative process, the LDA model is written as

$$p(\boldsymbol{w}, \boldsymbol{z}, \boldsymbol{\theta}, \boldsymbol{\phi}|\alpha, \gamma) = \prod_{t=1}^{T} \text{Dir}(\boldsymbol{\phi}_t; \gamma) \prod_{d=1}^{D} \text{Dir}(\boldsymbol{\theta}_d; \alpha) \prod_{i=1}^{N_d} \theta_{d,z_{di}} \phi_{w_{di},z_{di}} \tag{1.9}$$

After integrating out $\boldsymbol{z}$, LDA is actually a probabilistic version of PCA (or FA) which exploits the low-dimensional latent structure of the term frequency matrix. A natural extension of LDA is to capture the uncertainty of the number of topics by replacing Dirichlet process for Dirichlet Distribution [TJBB06].

1.3 Thesis Organization

The remaining Chapters are organized as follows:

In Chapter 2, beta process factor analysis (BPFA) [TJ07, PC09b] is considered for learning dictionaries for sparse image representations, with applications in compressive sensing (CS). A truncated beta-Bernoulli process is employed as a prior for learning the dictionary and image reconstruction. In addition, this non-parametric method naturally infers an appropriate dictionary size. The Dirichlet process (DP) [Fer73] and a probit stick-breaking process (PSBP) [RD09] are also considered to exploit structure within an image. The proposed method can learn a sparse dictionary *in situ*; training images may be exploited if available, but they are not required. Several example results are presented, with comparisons to other state-of-the-art approaches.

In Chapter 3, Nonparametric Bayesian (NPB) methods are employed to learn a statistical embedding, in which high-dimensional data are embedded in a low-dimensional latent space. A new nonlinear, reversible statistical embedding procedure is developed, generalizing spectral methods like diffusion maps [CL06] and Isomap [TdSL00]. The new statistical model also allows accurate estimation of latent-space dimensionality. Hierarchical Bayesian methods are also developed to learn a nonlinear dynamic model in the low-dimensional embedding space, allowing joint analysis of multiple types of dynamic data, sharing strength and inferring inter-relationships. In addition to *analyzing* dynamic data, the learned model also yields effective synthesis. Example results are presented for statistical embedding, latent-space dimensionality estimation, and analysis and synthesis of high-dimensional (dynamic) motion-capture data.

The problem of learning topic hierarchies from corpus is addressed in Chapter 4. We build a hierarchical tree-based topic model with two unique merits: i a Markovian

statistical relationship between subtopics is proposed, with the goal of re-using sub-topics deep in the tree; and (*ii*) if a word is used in a sub-topic at a particular scale, it is not re-used at any other scales. We also present a retrospective sampler to infer both the tree depth and width (the width is scale-dependent). Encouraging results are achieved on five data sets, with comparisons against recently developed related models.

Chapter 5 summarizes this thesis and its contributions. Several possible directions are also discussed for future work.

2

Bayesian Dictionary Learning for Compressive Sensing

2.1 Introduction

There has been significant recent interest in sparse image representations, in the context of denoising, interpolation [AEB06, EA06, MES08, MBPS09, MBP+08, MSE08, MBP+09, RPCL06], compressive sensing (CS) [CT06, DCS09], and classification [WYG+09]. All of these applications exploit the fact that images may often be sparsely represented in an appropriate dictionary. Most of the denoising, interpolation, and CS literature assumes "off-the-shelf" wavelet and DCT bases/dictionaries [JXC08], but recent research has demonstrated the significant utility of learning an often over-complete dictionary matched to the signals of interest (*e.g.*, images) [AEB06, EA06, MES08, MBPS09, MBP+08, MSE08, MBP+09, RPCL06, DCS09, RBL+07, BDE07].

Many of the existing methods for learning dictionaries are based on solving an optimization problem [AEB06, EA06, MES08, MBPS09, MBP+08, MSE08, MBP+09], in which one seeks to match the dictionary to the imagery of interest, while si-

multaneously encouraging a sparse representation. These methods tend to yield over-complete dictionaries, and have demonstrated state-of-the-art performance for denoising, interpolation, and inpainting. However, many existing algorithms for implementing such ideas also have some restrictions. For example, one must often assume access to the noise/residual variance, the size of the dictionary is set *a priori*, and a single ("point") estimate is learned.

To mitigate the aforementioned limitations, dictionary learning has recently been cast as a factor-analysis problem, with the factor loadings corresponding to the dictionary elements (atoms). Utilizing nonparametric Bayesian methods like the beta process (BP) [PC09b, ZCP+09, TJ07] and the Indian buffet process (IBP) [GG05, KG07], one may infer the number of factors (dictionary elements) needed to fit the data. The existing Bayesian variable selection algorithms [SK96, BD09] may also be used to select a subset of dictionary elements from a large number of potential ones. In these approaches, however, the weights corresponding to the unused dictionary elements are close to zero, not exactly zero. Further, one may place a prior on the noise or residual variance, with this inferred from the data [PC09b, ZCP+09]. An approximation to the full posterior may be manifested via Gibbs sampling, yielding an ensemble of dictionary representations. Recent research has demonstrated that an ensemble of representations can be better than a single expansion [EY10], with such an ensemble naturally manifested by statistical models as the one here described. Overall, the here proposed Bayesian framework provides a complementary and alternative framework with respect to the more standard variational formulations. These can also be interpreted via statistical models with solutions obtained via MAP estimation, *e.g.*, see [RS10] for an overview and new interpretation of this. Such probabilistic interpretations use models different than the ones here exploited, and as mentioned above, have to estimate critical parameters and produce single solutions.

In image analysis there is often additional information that may be exploited when learning dictionaries, with this well suited for Bayesian priors. For example, most natural images may be segmented, and it is probable that dictionary usage will be similar for regions within a particular segment class. To address this idea, we extend the model by employing a probit stick-breaking process (PSBP), with this a generalization of the Dirichlet process (DP) stick-breaking representation [Set94]. The model clusters the image patches, with each cluster corresponding to a segment type; the PSBP encourages proximate and similar patches to be included within the same segment type, thereby performing image segmentation and dictionary learning simultaneously.

As discussed, the proposed method is a natural tool for images denoising, applicable even the noise statistics are nonstationary and the nonstationary noise variance can be inferred within the analysis. The principal focus of this chapter, however, is on applying the algorithms to new compressive measurement techniques that have been developed recently. Specifically, we consider dictionary learning in the context of compressive sensing (CS) [CT06, Don06], in which the measurements correspond to projections of typical image pixels. We consider dictionary learning performed "offline" based on representative (training) images, with the learned dictionary applied within CS image recovery. We also consider the "online BP" case for which the underlying dictionary is learned simultaneously with inversion, with this related to "blind" CS [GE10]. Finally, we design the CS projection matrix to be matched to the learned dictionary (when this is done offline), and demonstrate as in [DCS09] that in practice this yields performance gains relative to conventional random CS projection matrices.

The remainder of this chapter is organized as follows. In Section 2.2 we briefly review the compressive sensing. The beta-Bernoulli process is discussed in Section 2.3, with relationships made with previous work in this area, including that based

on the Indian buffet process. The Dirichlet and probit stick-breaking processes are discussed in Section 2.4, and several example results are presented in Section 2.5. Conclusions and a discussion of future work are provided in Section 2.6, and details of the inference equations are summarized in Appendix A.

2.2 Compressive Sensing

We consider data samples that may be expressed in the form

$$\boldsymbol{x}_i = \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i \tag{2.1}$$

where $\boldsymbol{x}_i \in \mathbb{R}^P$, $\boldsymbol{\epsilon}_i \in \mathbb{R}^P$, and $\boldsymbol{w}_i \in \mathbb{R}^K$. The columns of the matrix $\mathbf{D} \in \mathbb{R}^{P\times K}$ represent the K components of a dictionary with which $\boldsymbol{x}_i$ is expanded. For our problem, the $\boldsymbol{x}_i$ will correspond to $B \times B$ pixel patches in an image [AEB06, EA06, MES08, MBPS09, MBP+08, ZCP+09]. The *set* of vectors $\{\boldsymbol{x}_i\}_{i=1,N}$ may be extracted from an image(s) of interest.

In many applications the total number of pixels $N \cdot P$ may be large. However, it is well known that compression algorithms may be used on $\{\boldsymbol{x}_i\}_{i=1,N}$ *after* the measurements have been performed, to significantly reduce the quantity of data that need be stored or communicated. This compression indicates that while the data dimensionality $N \cdot P$ may be large, the underlying information content may be relatively low. This has motivated the field of compressive sensing [CT06, Don06, DDT+08, SPB10], in which the total number of measurements performed may be much less than $N \cdot P$. Toward this end, researchers have proposed *projection* measurements of the form

$$\boldsymbol{y}_i = \boldsymbol{\Sigma}\boldsymbol{x}_i \tag{2.2}$$

where $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times P}$ and $\boldsymbol{y}_i \in \mathbb{R}^n$, ideally with $n \ll P$. The projection matrix $\boldsymbol{\Sigma}$ has traditionally been constituted randomly [CT06, Don06], with a binary or real alphabet (and $\boldsymbol{\Sigma}$ may also be a function of the specific patch, and generalized as $\boldsymbol{\Sigma}_i$). It is desirable that matrices $\boldsymbol{\Sigma}$ and $\mathbf{D}$ be as incoherent as possible.

The recovery of $\boldsymbol{x}_i$ from $\boldsymbol{y}_i$ is an ill-posed problem unless restrictions are placed on $\boldsymbol{x}_i$. We may exploit the same class of restrictions used in the denoising problem; specifically, the observed data satisfy $\boldsymbol{y}_i = \boldsymbol{\Phi}\boldsymbol{w}_i + \boldsymbol{\nu}_i$, with $\boldsymbol{\Phi} = \boldsymbol{\Sigma}\mathbf{D}$ and $\boldsymbol{\nu}_i = \boldsymbol{\Sigma}\boldsymbol{\epsilon}_i$, and with sparse $\boldsymbol{w}_i$. Note that the sparseness constraint implies that $\{\boldsymbol{w}_i\}_{i=1,N}$ (and hence $\{\boldsymbol{x}_i\}_{i=1,N}$) occupy a *nonlinear* subspace of $\mathbb{R}^P$.

In most applications of compressive sensing $\mathbf{D}$ is assumed known, corresponding to an orthonormal basis (*e.g.*, wavelets or a DCT) [CT06, Don06, JXC08]. However, such bases are not necessarily well matched to natural imagery, and it is desirable to consider design of dictionaries $\mathbf{D}$ for this purpose [DCS09]. One may even consider recovering $\{\boldsymbol{x}_i\}_{i=1,N}$ from $\{\boldsymbol{y}_i\}_{i=1,N}$ while simultaneously inferring $\mathbf{D}$. Thus, we again have a dictionary-learning problem, which may be coupled with optimization of the CS matrix $\boldsymbol{\Sigma}$, such that it is matched to $\mathbf{D}$ (defined by a low coherence between the rows of $\boldsymbol{\Sigma}$ and columns of $\mathbf{D}$ [CT06, Don06, JXC08]).

2.3 Sparse Factor Analysis with the Beta-Bernoulli Process

For the *compressive-sensing* application we observe $\boldsymbol{y}_i = \boldsymbol{\Sigma}(\mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i) = \boldsymbol{\Phi}\boldsymbol{w}_i + \boldsymbol{\nu}_i$, with $\boldsymbol{\Phi} = \boldsymbol{\Sigma}\mathbf{D}$ and $\boldsymbol{\nu}_i = \boldsymbol{\Phi}\boldsymbol{\epsilon}_i$. Our objective is to infer the underlying signal $\mathbf{D}\boldsymbol{w}_i$, with $\boldsymbol{w}_i$ assumed sparse; we generally wish to simultaneously infer $\mathbf{D}$ and $\{\boldsymbol{w}_i\}_{i=1,N}$. To address this problem, we consider a statistical model for $\boldsymbol{x}_i = \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i$, placing Bayesian priors on $\mathbf{D}$, $\boldsymbol{w}_i$ and $\boldsymbol{\epsilon}_i$; the way the model is used is modified slightly for each specific application.

2.3.1 Beta-Bernoulli process for active-set selection

Let the binary vector $\boldsymbol{z}_i \in \{0,1\}^K$ denote which columns of $\mathbf{D}$ are used for representation of $\boldsymbol{x}_i$; if a particular component of $\boldsymbol{z}_i$ is equal to one, then the corresponding column of $\mathbf{D}$ is used in the representation of $\boldsymbol{x}_i$. Hence, for the data $\{\boldsymbol{x}_i\}_{i=1,N}$ there is an associated set of latent binary vectors $\{\boldsymbol{z}_i\}_{i=1,N}$, and the beta-Bernoulli process

provides a convenient prior for these vectors [PC09b, ZCP+09, TJ07]. Specifically, consider the model

$$\begin{aligned} \boldsymbol{z}_i &\sim \prod_{k=1}^{K} \text{Bernoulli}(\pi_k) \\ \boldsymbol{\pi} &\sim \prod_{k=1}^{K} \text{Beta}(a/K, b(K-1)/K) \end{aligned} \tag{2.3}$$

where π_k is the kth component of $\boldsymbol{\pi}$.

Considering the limit $K \to \infty$, and after integrating out $\boldsymbol{\pi}$, the draws of $\{\boldsymbol{z}_i\}_{i=1,N}$ may be constituted as follows. For each $\boldsymbol{z}_i$, draw $c_i \sim \text{Poisson}(\frac{a}{b+i-1})$ and define $C_i = \sum_{j=1}^{i} c_j$, with $C_0 = 0$. Let z_{ik} represent the kth component of $\boldsymbol{z}_i$, and $z_{ik} = 0$ for $k > C_i$. For $k = 1, \ldots, C_{i-1}$, $z_{ik} \sim \text{Bernoulli}(\frac{n_{ik}}{b+i-1})$, where $n_{ik} = \sum_{j=1}^{i-1} z_{jk}$ (n_{ik} represents the total number of times the kth component of $\{\boldsymbol{z}_j\}_{j=1,i-1}$ is one). For $k = C_{i-1} + 1, \ldots, C_i$, we set $z_{ik} = 1$. Note that as $a/(b+i-1)$ becomes small, with increasing i, it is probable that c_i will be small. Hence, with increasing i, the number of new non-zero components of $\boldsymbol{z}_i$ diminishes. Further, as a consequence of Bernoulli($\frac{n_{ik}}{b+i-1}$), when a particular component of the vectors $\{\boldsymbol{z}_j\}_{j=1,i-1}$ is frequently one, it is more probable that it will be one for subsequent $\boldsymbol{z}_j$, $j \geq i$. When $b = 1$ this construction for $\{\boldsymbol{z}_i\}_{i=1,N}$ corresponds to the Indian buffet process [GG05].

Since $\boldsymbol{z}_i$ defines which columns of $\mathbf{D}$ are used to represent $\boldsymbol{x}_i$, (2.3) imposes that it is probable that some columns of $\mathbf{D}$ are used repeatedly among the set $\{\boldsymbol{x}_i\}_{i=1,N}$, while other columns of $\mathbf{D}$ may be more specialized to particular $\boldsymbol{z}_i$. As demonstrated below, this has been found to be a good model when $\{\boldsymbol{x}_i\}_{i=1,N}$ are patches of pixels extracted from natural images.

2.3.2 Full hierarchical model

The hierarchical form of the model may now be expressed as

$$\begin{aligned} \boldsymbol{x}_i &= \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i \\ \boldsymbol{w}_i &= \boldsymbol{z}_i \odot \boldsymbol{s}_i \\ \boldsymbol{d}_k &\sim \mathcal{N}(0, P^{-1}\mathbf{I}_P) \\ \boldsymbol{s}_i &\sim \mathcal{N}(0, \gamma_s^{-1}\mathbf{I}_K) \\ \boldsymbol{\epsilon}_i &\sim \mathcal{N}(0, \gamma_\epsilon^{-1}\mathbf{I}_P) \end{aligned} \tag{2.4}$$

where $\boldsymbol{d}_k$ represents the kth component of $\mathbf{D}$, $\circ$ represents the pointwise or Hadamard vector product, $\mathbf{I}_P$ represents a $P \times P$ identity matrix, and $\{\boldsymbol{z}_i\}_{i=1,N}$ are drawn as in (2.3). Conjugate hyperpriors $\gamma_s \sim \text{Gamma}(c, d)$ and $\gamma_\epsilon \sim \text{Gamma}(e, f)$ are also imposed. The construction in (2.4), and with the prior in (2.3) for $\{\boldsymbol{z}_i\}_{i=1,N}$, is henceforth referred to as the beta process factor analysis (BPFA) model.

Note that we impose independent Gaussian *priors* for $\boldsymbol{d}_k$, $\boldsymbol{s}_i$ and $\boldsymbol{\epsilon}_i$ for modeling convenience (conjugacy of consecutive terms in the hierarchical model). However, the inferred *posterior* for these terms is generally *not* independent or Gaussian. The independent priors essentially impose prior information about the *marginals* of the posterior of each component, while the inferred posterior accounts for statistical dependence as reflected in the data. To make connections of this model to more-typical optimization-based approaches [MBPS09, MBP+08], note that the negative logarithm of the posterior density function is

$$\begin{aligned} -\log\ p(\boldsymbol{\Theta}|\mathcal{D}, \mathcal{H}) &= \frac{\gamma_\epsilon}{2}\sum_{i=1}^{N}\|\boldsymbol{x}_i - \mathbf{D}(\boldsymbol{s}_i \circ \boldsymbol{z}_i)\|_2^2 + \frac{P}{2}\sum_{k=1}^{K}\|\boldsymbol{d}_k\|_2^2 + \frac{\gamma_s}{2}\sum_{i=1}^{N}\|\boldsymbol{s}_i\|_2^2 \\ &- \log f_{Beta-Bern}(\{\boldsymbol{z}_i\}_{i=1}^N; \mathcal{H}) - \log \text{Gamma}(\gamma_\epsilon|\mathcal{H}) \\ &- \log \text{Gamma}(\gamma_s|\mathcal{H}) + Const. \end{aligned} \tag{2.5}$$

where $\mathcal{D} = \{\boldsymbol{x}_i\}_{i=1,N}$, $f_{Beta-Bern}(\{\boldsymbol{z}_i\}_{i=1}^N; \mathcal{H})$ represents the beta-Bernoulli process prior in (2.3), $\boldsymbol{\Theta}$ represents all unknown model parameters, and $\mathcal{H}$ represents model hyper-parameters (*i.e.*, a, b, c, d, e and f). Therefore, the typical ℓ_2 constraints [MBPS09, MBP$^+$08] on the dictionary elements $\boldsymbol{d}_k$ and on the non-zero weights $\boldsymbol{s}_i$ correspond here to the Gaussian priors employed in (2.4). However, rather than an employing an ℓ_1 constraint [MBPS09, MBP$^+$08] to impose sparseness on $\boldsymbol{w}_i$, we employ the beta-Bernoulli process and $\boldsymbol{w}_i = \boldsymbol{s}_i \circ \boldsymbol{z}_i$. The beta-Bernoulli process imposes that the binary $\boldsymbol{z}_i$ should be sparse, *and* that there should be a relatively consistent (re)use of dictionary elements across the image, thereby imposing self-similarity. Further, and perhaps most importantly, we do *not* constitute a point estimate, as one would do if a single $\boldsymbol{\Theta}$ was sought to maximize (2.5). We rather estimate the full posterior density $p(\boldsymbol{\Theta}|\mathcal{D}, \mathcal{H})$, implemented via Gibbs sampling. A significant advantage of the hierarchical construction in (2.4) is that each Gibbs update equation is analytic, with detailed update equations provided in Appendix A. Note that consistent use of atoms is encouraged because the active sets are defined by the binary vectors $\{\boldsymbol{z}_i\}_{i=1,\ldots,N}$, and these are all drawn from a shared probability vector $\boldsymbol{\pi}$; this is distinct from drawing the active sets i.i.d. from a Laplacian prior. Further, the beta-Bernoulli prior imposes that many components of $\boldsymbol{w}_i$ are exactly zero, while with a Laplacian prior many components are small but not exactly zero.

2.4 Dirichlet and Probit Stick-Breaking Processes

2.4.1 Dirichlet process

As discussed in the previous section, in many applications it is expected that the data $\{\boldsymbol{x}_i\}_{i=1,N}$ may cluster, and it is of interest to infer this clustering nonparametrically (*i.e.*, to infer the number of clusters and their composition from the data). The imposition of such prior knowledge may improve the quality of the inversion for $\{\boldsymbol{x}_i\}_{i=1,N}$ based upon incomplete measurements. The Dirichlet process (DP) [Fer73]

constitutes a popular means of performing such nonparametric clustering. A random draw from a DP, $G \sim \mathrm{DP}(\alpha G_0)$, with precision $\alpha \in \mathbb{R}^+$ and "base" measure G_0, may be constituted via the stick-breaking construction [Set94]

$$G = \sum_{l=1}^{\infty} \beta_l \delta_{\theta_l^*} \quad , \quad \theta_l^* \sim G_0 \tag{2.6}$$

where $\beta_l = V_l \prod_{h=1}^{l-1}(1 - V_h)$ and $V_h \sim \mathrm{Beta}(1, \alpha)$. The β_l may be viewed as a sequence of fractional breaks from a "stick" of original length one, where the fraction of stick broken off on break l is V_l. The θ_l^* are model parameters, associated with the lth data cluster. For our problem it has proven effective to set $G_0 = \prod_{k=1}^{K} \mathrm{Beta}(a/K, b(K-1)/K)$ analogous to (2.3), and hence $G = \sum_{l=1}^{\infty} \beta_l \delta_{\boldsymbol{\pi}_l^*}$. The $\boldsymbol{\pi}_l^*$, drawn from G_0, correspond to distinct probability vectors for using the K dictionary elements (columns of $\mathbf{D}$). A separate sparse binary vector $\boldsymbol{z}_i$ is drawn for each sample $\boldsymbol{x}_i$, as $\boldsymbol{z}_i \sim \prod_{k=1}^{K} \mathrm{Bernoulli}(\pi_{ik})$, with π_{ik} the kth component of $\boldsymbol{\pi}_i$. In practice we truncate the infinite sum for G to N_L elements, and impose $V_{N_L} = 1$, such that $\sum_{l=1}^{N_L} \beta_l = 1$. A (conjugate) gamma prior is placed on the DP parameter α.

We may view this DP construction as an "Indian buffet franchise", generalizing the Indian buffet analogy [GG05]. Specifically, there are N_L Indian buffet restaurants; each restaurant is composed of the same "menu" (columns of $\mathbf{D}$), and is distinguished by different probabilities for selecting menu items. The "customers" $\{\boldsymbol{x}_i\}_{i=1,N}$ cluster based upon which restaurant they go to. The $\{\boldsymbol{\pi}_l^*\}_{l=1,N_L}$ represent the probability of using each column of $\mathbf{D}$ in the respective N_L different buffets. The data $\{\boldsymbol{x}_i\}_{i=1,N}$ cluster themselves among the different restaurants in a manner that is consistent with the characteristics of the data, with the model also simultaneously learning the dictionary/menu $\mathbf{D}$. Note that we typically make the truncation N_L large, and the posterior distribution infers the number of clusters actually needed to support the

data, as represented by how many β_l are of significant value. The model in (2.4), with the above DP construction for $\{\boldsymbol{z}_i\}_{i=1,N}$, is henceforth referred to as DP-BPFA. The variational Bayesian inference for DP-BPFA is also provided in Appendix A.

2.4.2 Probit stick-breaking process

The DP yields a clustering of $\{\boldsymbol{x}_i\}_{i=1,N}$, but it does not account for our knowledge of the location of each patch within the image. It is natural to expect that if $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i'}$ are proximate then they are likely to be constituted in terms of similar columns of $\mathbf{D}$. To impose this information, we employ the probit stick-breaking process (PSBP). A *logistic* stick-breaking process is discussed in detail in [RDC11]. We employ the closely related probit version here because it may be easily implemented in a Gibbs sampler. We note that while the method in [RDC11] is related to that discussed below, in [RDC11] the concepts of learned dictionaries and beta-Bernoulli priors were not considered.

We augment the data as $\{\boldsymbol{x}_i, \boldsymbol{r}_i\}_{i=1,N}$, where $\boldsymbol{x}_i$ again represents pixel values from the ith image patch, and $\boldsymbol{r}_i \in \mathbb{R}^2$ represents the two-dimensional location of each patch. We wish to impose that proximate patches are more likely to be composed of the same or similar columns of $\mathbf{D}$. In the PSBP construction, all aspects of (2.4) are retained, except for the manner in which $\boldsymbol{z}_i$ are constituted. Rather than drawing a single K-dimensional vector of probabilities $\boldsymbol{\pi}$ as in (2.3), we draw a *library* of such vectors:

$$\boldsymbol{\pi}_l^* \sim \prod_{k=1}^{K} \text{Beta}(a/K, b(K-1)/K) \quad , \quad l = 1, \ldots, N_L \tag{2.7}$$

and each $\boldsymbol{\pi}_l^*$ is associated with a particular segment in the image. One $\boldsymbol{\pi}_i$ is associated with location $\boldsymbol{r}_i$, and drawn

$$\boldsymbol{\pi}_i \sim \sum_{l=1}^{N_L} \beta_l(\boldsymbol{r}_i)\delta_{\boldsymbol{\pi}_l^*} \tag{2.8}$$

with $\sum_{l=1}^{N_L} \beta_l(\boldsymbol{r}_i) = 1$ for all $\boldsymbol{r}_i$, and $\delta_{\boldsymbol{\pi}_l^*}$ represents a point measure concentrated at $\boldsymbol{\pi}_l^*$. Once $\boldsymbol{\pi}_i$ is associated with a particular $\boldsymbol{x}_i$, the corresponding binary vector $\boldsymbol{z}_i$ is drawn as in the first line of (2.3). Note that the distinction between DP and PSBP is that in the former the mixture weights $\{\beta_l\}_{l=1,N_L}$ are independent of spatial position $\boldsymbol{r}$, while the latter explicitly utilizes $\boldsymbol{r}$ within $\{\beta_l(\boldsymbol{r})\}_{l=1,N_L}$ (and below we impose that $\beta_l(\boldsymbol{r})$ changes smoothly with $\boldsymbol{r}$).

The space-dependent weights are constructed as $\beta_l(\boldsymbol{r}) = V_l(\boldsymbol{r}) \prod_{h=l}^{l-1}[1 - V_h(\boldsymbol{r})]$ where $0 < V_l(\boldsymbol{r}) < 1$ constitute space-dependent probabilities. We set $V_{N_L} = 1$, and for $l \leq N_L - 1$ the V_l are space-dependent probit functions:

$$V_l(\boldsymbol{r}) = \int_{-\infty}^{g_l(\boldsymbol{r})} dx \mathcal{N}(x|0,1) \tag{2.9}$$

$$g_l(\boldsymbol{r}) = \zeta_{l0} + \sum_{i=1}^{N} \zeta_{li} \mathcal{K}(\boldsymbol{r}, \boldsymbol{r}_i; \psi_l) \tag{2.10}$$

where $\mathcal{K}(\boldsymbol{r}, \boldsymbol{r}_i; \psi_l)$ is a kernel characterized by parameter ψ_l and $\{\zeta_{li}\}_{i=0,N}$ are a *sparse* set of real numbers. To implement the sparseness on $\{\zeta_{li}\}_{i=0,N}$, within the prior $\zeta_{li} \sim \mathcal{N}(0, \alpha_{li}^{-1})$, and $\alpha_{li} \sim \text{Gamma}(a_0, b_0)$, with (a_0, b_0) set to favor most α_{li} being large (if α_{li} is large, a draw $\mathcal{N}(0, \alpha_{li}^{-1})$ is likely to be near zero, such that most $\{\zeta_{li}\}_{i=0,N}$ are near zero). This sparseness-promoting construction is the same as that employed in the relevance vector machine (RVM) [Tip01]. We here utilize a radial basis function (RBF) kernel $\mathcal{K}(\boldsymbol{r}, \boldsymbol{r}_i; \psi_l) = \exp[-\|\boldsymbol{r}_i - \boldsymbol{r}\|_2/\psi_l]$.

Each $g_l(\boldsymbol{r})$ is encouraged to only be defined by a small set of localized kernel functions, and via the probit link function $\int_{-\infty}^{g_l(\boldsymbol{r})} dx \mathcal{N}(x|0,1)$ the probability $V_l(\boldsymbol{r})$ is characterized by localized segments over which the probability $V_l(\boldsymbol{r})$ is contiguous and smoothly varying. The $V_l(\boldsymbol{r})$ constitute a space-dependent stick-breaking process. Since $V_{N_L} = 1$, $\sum_{l=1}^{N_L} \beta_l(\boldsymbol{r}) = 1$ for all $\boldsymbol{r}$.

The PSBP model is relatively simple to implement within a Gibbs sampler. For

example, as indicated above, sparseness on ζ_{li} is imposed as in the RVM, and the probit link function is simply implemented within a Gibbs sampler (which is why it was selected, rather than a logistic link function). Finally, we define a finite set of possible kernel parameters $\{\psi_j\}_{j=1,N_p}$, and a multinomial prior is placed on these parameters, with the multinomial probability vector drawn from a Dirichlet distribution [RDC11] (each of the $g_l(\boldsymbol{r})$ draws a kernel parameter from $\{\psi_j\}_{j=1,N_p}$). The model in (2.4), with the PSBP construction for $\{\boldsymbol{z}_i\}_{i=1,N}$, is henceforth referred to as PSBP-BPFA.

2.4.3 Discussion of sparseness-imposing priors

The basic BPFA model is summarized in (2.4), and three related priors have been developed for the sparse binary vectors $\{\boldsymbol{z}_i\}_{i=1,N}$: (i) the basic truncated beta-Bernoulli process in (2.3), (ii) a DP-based clustering of the underlying $\{\boldsymbol{\pi}_i\}_{i=1,N}$, and (iii) a PSBP clustering of $\{\boldsymbol{\pi}_i\}_{i=1,N}$ that exploits knowledge of the location of the image patches. For (ii) and (iii), the $\boldsymbol{x}_i$ within a particular cluster have *similar* $\boldsymbol{z}_i$, rather than exactly the same binary vector; we also considered the latter, but this worked less well in practice. As discussed further when presenting results, (ii) and (iii) yield marked improvements in image-recovery accuracy relative to (i). In anticipation of these results, we provide a further discussion of the three priors on $\{\boldsymbol{z}_i\}_{i=1,N}$.

For the CS problem, we measure $\boldsymbol{y}_i = \boldsymbol{\Sigma}\boldsymbol{x}_i$, and therefore each of the n measurements associated with each image patch ($\boldsymbol{\Sigma} \in \mathbb{R}^{n\times P}$) loses the original pixels in $\boldsymbol{x}_i$ (the projection matrix $\boldsymbol{\Sigma}$ may also change with each patch, denoted $\boldsymbol{\Sigma}_i$). In this case, one cannot consider all possible shifts of the patches, as the patches are predefined and fixed in the CS measurement. Therefore, for CS imposition of the clustering behavior via DP or PSBP provides important information, yielding state-of-the-art CS-recovery results.

2.5 Example Results

We compare the performance of BPFA(BP), DP-BPFA(DP BP), PSBP-BPFA(PSBP BP) and online BP to six recently developed CS reconstruction algorithms: basis pursuit [CDS99], Bayesian compressive sensing (BCS) [JXC08], fast-BCS [JXC08], orthogonal matching pursuit (OMP) [TG07], stagewise orthogonal matching pursuit (STOMP) [DTDS06], Lasso-modified least angle regression (LARS/Lasso) [EHJT04]. For the basis pursuit implementations, we used solvers from the ℓ_1-*Magic* toolbox[1]; for the OMP, STOMP and LARS/Lasso algorithms, we use the solvers `SolveOMP`, `SolveStOMP`, and `SolveLasso`, respectively, from the *SparseLab* toolbox[2]; for the BCS and fast-BCS, we use the packages `bcs_vb` and `bcs_ver0.1` from the BCS website [3]. All software are written in MATLAB^{TM}.

2.5.1 Parameter settings

For all BPFA, DP-BPFA and PSBP-BPFA computations, the dictionary truncation level was set at $K = 256$ based on the size of the image. Not all K dictionary elements are used in the model; the truncated beta-Bernoulli process infers the subset of dictionary elements employed to represent the data $\{\boldsymbol{x}_i\}_{i=1,N}$. The number of DP and PSBP sticks was set at $N_L = 20$. The library of PSBP parameters is defined as in [RDC11]. The hyperparameters within the gamma distributions were set as $c = d = e = f = 10^{-6}$, as is typically done in models of this type [Tip01] (the same settings were used for the gamma prior for the DP precision parameter α). The beta-distribution parameters are set as $a = 10$ and $b = 1$ for the dictionary learning or $a = n$ and $b = 1$ for the compressive sensing. None of these parameters have been optimized or tuned. When performing inference, all parameters are initialized

[1] $http://www.acm.caltech.edu/l1magic/$

[2] $http://sparselab.stanford.edu/$

[3] $http://www.ece.duke.edu/\sim lihan/cs/$

randomly (as a draw from the associated prior) or based on the SVD of the image under test.

2.5.2 Comparisons to other compressive sensing algorithms

We consider a CS example in which the image is divided into 8×8 patches, with these constituting the underlying data $\{\boldsymbol{x}_i\}_{i=1,N}$ to be inferred. For each of the N blocks, a vector of CS measurements $\boldsymbol{y}_i = \boldsymbol{\Sigma}_i \boldsymbol{x}_i$ is measured, where the number of projections per patch is n, and the total number of CS projections is $n \cdot N$. In our first examples the elements of $\boldsymbol{\Sigma}_i$ are constructed randomly, as draws from $\mathcal{N}(0, 1)$; many other random projection classes may be considered [Bar07] (and below we also consider optimized projections $\boldsymbol{\Sigma}_i$, matched to the dictionary $\mathbf{D}$). Each $\boldsymbol{x}_i$ is assumed represented in terms of a dictionary $\boldsymbol{x}_i = \mathbf{D}\boldsymbol{w}_i + \boldsymbol{\epsilon}_i$, and three constructions for $\mathbf{D}$ were considered: (i) a DCT expansion; (ii) learning of $\mathbf{D}$ using BPFA, using training images; (iii) using the BPFA, DP-BPFA and PSBP-BPFA to perform *joint* CS inversion and learning of $\mathbf{D}$. For (ii), the training data consisted of 4000 8×8 patches chosen at random from 100 training images selected from the Microsoft database[4]. The dictionary was set to $K = 256$, and the offline beta process inferred a dictionary of size $M = 237$.

Representative CS reconstruction results are shown in Figure 2.1 based upon a DCT dictionary, for a grey-scale version of the "castle" image. The results in Figure 2.2 are based on a learned dictionary; except for the "online BP" results, all of these results employ the same dictionary $\mathbf{D}$ learned off-line as above, and the algorithms are distinguished by different ways of estimating $\{\boldsymbol{w}_i\}_{i=1,N}$. A range of CS-inversion algorithms are considered from the literature, and several BPFA-based constructions are considered as well for CS inversion. The online BPFA results (with no training data) are quite competitive with those based on a dictionary learned

[4] http://research.microsoft.com/en-us/projects/objectclassrecognition

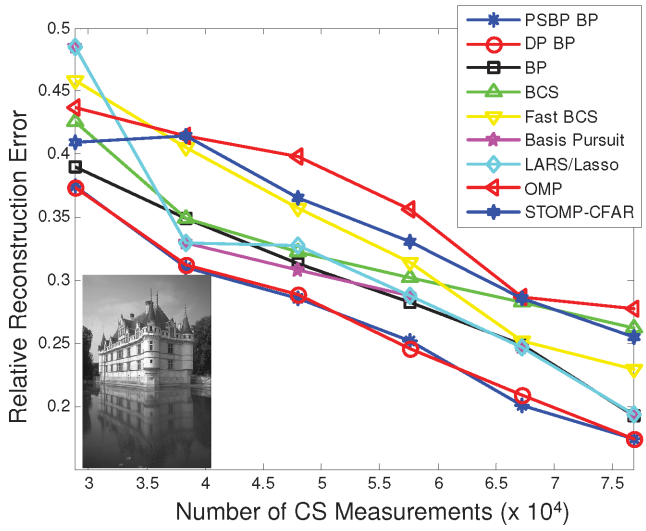


FIGURE 2.1: Compressive sensing (CS) results on grey-scale Castle image, based on a DCT dictionary $\mathbf{D}$. The CS projection matrix $\mathbf{\Sigma}_i$ is constituted randomly, with elements drawn iid from $\mathcal{N}(0,1)$. Results are shown using the DP-BPFA and PSBP-BPFA models in Section 2.4. Comparisons are also made with several CS inversion algorithms from the literature.

off-line. Note that results based on a learned dictionary are markedly better than those based on the DCT; similar results were achieved when the DCT was replaced by a wavelet representation. For the DCT-based results, note that the DP-BPFA and PSBP-BPFA CS inversion results are significantly better than those of all other CS inversion algorithms. The average performance of all algorithms is reported in Table 2.1. The results are based on 100 test images from the aforementioned Microsoft database. Note that the performance of DP-BPFA and PSBP-BPFA CS inversion is still much better than that of all other CS inversion algorithms for both DCT and learned dictionary.

In all previous results the projection matrix $\mathbf{\Sigma}$ was constituted randomly. We now consider a simple means of matching $\mathbf{\Sigma}$ to a $\mathbf{D}$ learned offline, based upon representative training images. Assume a learned $\mathbf{D} \in \mathbb{R}^{P\times K}$, with $K > P$, which may be represented via SVD as $\mathbf{D} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$; $\mathbf{U} \in \mathbb{R}^{P\times P}$ and $\mathbf{V} \in \mathbb{R}^{K\times P}$ are each

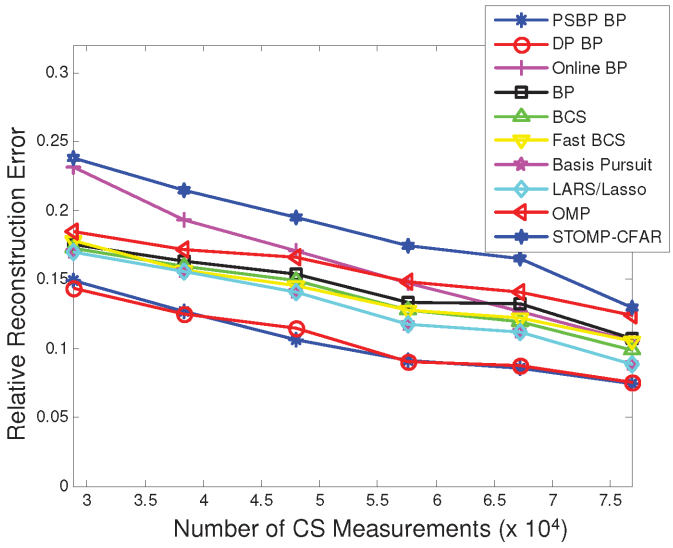


FIGURE 2.2: Compressive sensing (CS) results on grey-scale Castle image, based on a learned dictionary $\mathbf{D}$. The online BP results employ BPFA to learn $\mathbf{D}$ and do CS inversion jointly. All other results are based upon a learned $\mathbf{D}$ with learning performed offline using distinct training images. The CS projection matrix $\mathbf{\Sigma}_i$ is constituted randomly, with elements drawn iid from $\mathcal{N}(0,1)$. Results are shown using the PSBP-BPFA and DP-BPFA models in Section 2.4. Comparisons are also made with several CS inversion algorithms from the literature.

composed of orthonormal columns, and $\mathbf{\Lambda}$ is a $P \times P$ diagonal matrix. The columns of $\mathbf{U}$ span the linear subspace of $\mathbb{R}^P$ in which the columns of $\mathbf{D}$ reside. Further, since the columns of $\mathbf{D}$ are generally *not* orthonormal, each column of $\mathbf{D}$ is "spread out" when expanded in the columns of $\mathbf{U}$. Therefore, one expects that $\mathbf{U}$ and $\mathbf{D}$ are incoherent. Hence, a simple means of matching CS projections to the data is to define the rows of $\mathbf{\Sigma}$ in terms of randomly selected columns of $\mathbf{U}$. This was done in Figure 2.3 for the grey-scale "castle" image, using the same learned dictionary as considered in Figure 2.2. It is observed that this procedure yields a marked improvement in CS recovery accuracy, for all CS inversion algorithms considered.

Table 2.1: Mean and standard deviation of the reconstruction error for DCT and learned dictionary D for each class in the Microsoft database. The bold numbers represent the best two among all the CS algorithms under comparison.

(a) DCT, number of measurements $n = 25$

Class \ Algorithm		DP BP	PSBP BP	BP	OMP	StOMP -CFAR	VB BCS	Fast BCS	LARS/ Lasso	Basis Pursuit
Buildings	mean	0.1025	**0.1010**	0.1317	0.1383	0.1570	0.1305	0.1486	0.1302	0.1344
	std	0.0421	**0.0374**	0.0364	0.0375	0.0499	0.0367	0.0480	0.0347	0.0347
Cows	mean	**0.0638**	0.0670	0.0905	0.0962	0.1120	0.0895	0.1045	0.0893	0.0922
	std	0.0386	**0.0329**	0.0409	0.0429	0.0437	0.0404	0.0537	0.0402	0.0402
Flowers	mean	0.0932	**0.0887**	0.1018	0.1145	0.1443	0.0992	0.1142	0.1044	0.1083
	std	0.0397	**0.0354**	0.0359	0.0350	0.0436	0.0310	0.0361	0.0330	0.0330
Office	mean	0.1069	**0.1033**	0.1171	0.1262	0.1600	0.1148	0.1282	0.1183	0.1203
	std	**0.0312**	0.0338	0.0336	0.0370	0.0430	0.0338	0.0368	0.0367	0.0367
Urban	mean	**0.1012**	0.1051	0.1218	0.1243	0.1600	0.1162	0.1382	0.1144	0.1182
	std	0.0175	**0.0170**	0.0214	0.0216	0.0237	0.0207	0.0190	0.0214	0.0214

(b) Learned Dictionary D, $n = 25$

Class \ Algorithm		DP BP	PSBP BP	BP	OMP	StOMP -CFAR	VB BCS	Fast BCS	LARS/ Lasso	Basis Pursuit	Online BP
Buildings	mean	0.0616	**0.0605**	0.1017	0.1163	0.1396	0.1151	0.1033	0.1289	0.0973	0.1213
	std	**0.0241**	0.0244	0.0379	0.0451	0.0369	0.0421	0.0413	0.0641	0.0358	0.0390
Cows	mean	0.0420	**0.0419**	0.0729	0.0855	0.0972	0.0801	0.0739	0.0720	0.0702	0.0895
	std	**0.0262**	0.0273	0.0389	0.0477	0.0418	0.0419	0.0405	0.0376	0.0358	0.0378
Flowers	mean	0.0391	**0.0390**	0.0818	0.0942	0.1185	0.0923	0.0831	0.1045	0.0809	0.0995
	std	0.0255	**0.0236**	0.0410	0.0510	0.0344	0.0449	0.0437	0.0586	0.0397	0.0330
Office	mean	**0.0399**	0.0408	0.0842	0.0855	0.1301	0.0865	0.0755	0.0934	0.0758	0.1047
	std	**0.0095**	0.0100	0.0329	0.0194	0.0356	0.0176	0.0175	0.0258	0.0174	0.0205
Urban	mean	0.0625	**0.0618**	0.1083	0.1198	0.1269	0.1174	0.1082	0.1074	0.0989	0.1222
	std	0.0111	**0.0100**	0.0192	0.0196	0.0216	0.0177	0.0190	0.0174	0.0146	0.0180

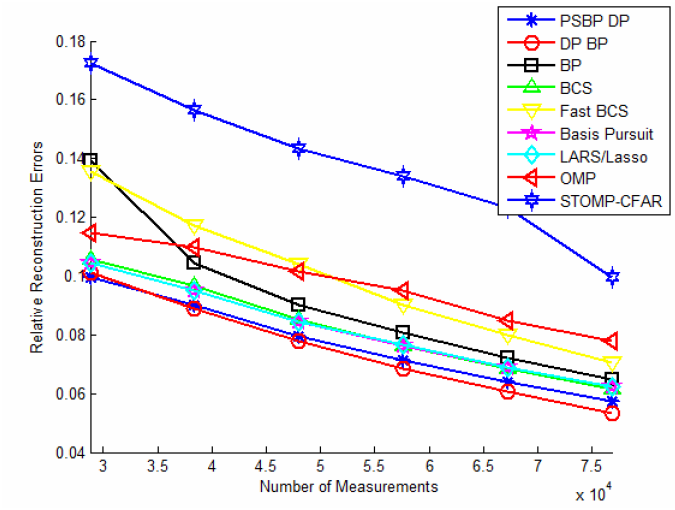


FIGURE 2.3: Compressive sensing (CS) results on grey-scale Castle image, based on a learned dictionary $\mathbf{D}$ (learning performed offline, using distinct training data). The projection matrix $\mathbf{\Sigma}$ is matched to $\mathbf{D}$, based upon an SVD of $\mathbf{D}$.

2.6 Conclusions

The truncated beta-Bernoulli process has been employed to learn dictionaries matched to image patches $\{\boldsymbol{x}_i\}_{i=1,N}$ extracted from imagery. The basic nonparametric Bayesian model is termed a beta process factor analysis (BPFA) framework, and extensions have also been considered. Specifically, the Dirichlet process (DP) has been employed to cluster the $\{\boldsymbol{x}_i\}_{i=1,N}$, encouraging similar dictionary-element usage within respective clusters. Further, the probit stick-breaking process (PSBP) has been used to impose that proximate patches are more likely to be clustered similarly (imposing that they are more probable to employ similar dictionary elements). All inference has been performed by a Gibbs sampler or variational Bayesian analysis, with analytic update equations. The BPFA, DP-BPFA and PSBP-BPFA have been applied to dictionaries learning for compressive sensing and also CS inversion. While training data may be used to initialize the dictionary learning, this is not needed, and the BPFA results are highly competitive even based upon random initializations. In the

context of compressive sensing, the DP-BPFA and PSBP-BPFA results are state of the art, significantly better than existing published methods. Finally, based upon the learned dictionary, a simple method has been constituted for optimizing the CS projections.

3

Hierarchical Bayesian Embeddings for Analysis and Synthesis of Dynamic Data

3.1 Introduction

The problem of inferring a low-dimensional latent space for representation of high-dimensional data has attracted significant attention for many years. The problem may be framed as follows. Given $\{\boldsymbol{y}_i\}_{i=1,N}$, with $\boldsymbol{y}_i \in \mathbb{R}^D$ for large D, we wish to learn a mapping or embedding that yields corresponding $\{\boldsymbol{x}_i\}_{i=1,N}$ with $\boldsymbol{x}_i \in \mathbb{R}^d$, ideally with $d \ll D$. If the data $\{\boldsymbol{y}_i\}_{i=1,N}$ are used to define the columns of the matrix $\boldsymbol{Y} \in \mathbb{R}^{D\times N}$, with $\boldsymbol{X} \in \mathbb{R}^{d\times N}$ similarly defined by $\{\boldsymbol{x}_i\}_{i=1,N}$, this mapping may be denoted concisely as $\boldsymbol{Y} \rightarrow \boldsymbol{X}$. We often desire the following properties of such an embedding: (i) an ability to accurately estimate an appropriate embedding dimension d, defined ideally by the dimensionality of the subspace of $\mathbb{R}^D$ in which $\boldsymbol{Y}$ resides (depending on the properties of $\boldsymbol{Y}$, the dimension d may vary across the support of the data); (ii) an ability to embed a new sample $\boldsymbol{y}_{N+1}$ without having to first return to the original training data $\{\boldsymbol{y}_i\}_{i=1,N}$; (iii) an efficient means of mapping from the embedding in $\mathbb{R}^d$ back to the space of the original data in $\mathbb{R}^D$;

and (iv) retention within the embedding space of relative distances between sample pairs, with the appropriate distance (*e.g.*, Euclidian or geodesic along a manifold) depending on the application.

The challenge posed by (i) is of longstanding interest; see [LB04] and the references therein. We develop a new approach to this problem, it providing excellent performance relative to existing methods, as demonstrated in several examples. The goal defined by (ii) has been a challenge in previous work, with most existing methods [CL06, Law05, DM05] requiring $\{\boldsymbol{y}_i\}_{i=1,N}$ when embedding a new $\boldsymbol{y}_{N+1}$. Concerning property (iii), we are motivated by the goal of analyzing *dynamic* high-dimensional data, in which the indices of $\{\boldsymbol{y}_i\}_{i=1,N}$ correspond to time. Our objective is to perform analysis of the dynamic behavior in the low-dimensional embedding space, as such analysis is significantly simpler in $\mathbb{R}^d$ than in $\mathbb{R}^D$, when $d \ll D$, particularly for a limited number of samples N. Additionally, after learning a dynamic model in the embedding space, we wish to use it to *synthesize* dynamic data, with related work considered in [THR07, WFH08, LM07, GMHP04]. The synthesized data will be viewed in the original space $\mathbb{R}^D$, and therefore we also require the ability to perform a reverse mapping from the dynamic data in the embedding space $\boldsymbol{X}$ back to the space of $\boldsymbol{Y}$, this motivating property (iii).

The distinction between different embedding methods is defined by the properties that one wishes to preserve about $\boldsymbol{Y}$ in the (compressed) form represented by $\boldsymbol{X}$. For example, a simple method is to perform a singular value decomposition (SVD) of $\boldsymbol{Y}$, with the columns of $\boldsymbol{X}$ defined by the weights on the d principal components [Ach03]. This mapping is (approximately) reversible, *i.e.*, we may approximately achieve $\boldsymbol{Y} \leftarrow \boldsymbol{X}$ in terms of a rank-d expansion using the d principal vectors, and the Frobenius norm between the original $\boldsymbol{Y}$ and its reconstruction is the minimum that may be achieved by any rank-d expansion. However, an SVD-based representation essentially assumes a Gaussian model for the data (the principal vectors are

the eigenvectors of the empirical covariance matrix, $\frac{1}{N}\boldsymbol{Y}\boldsymbol{Y}^T$), which may not be applicable in general. However, as discussed below, a *mixture* of low-rank Gaussians is often more appropriate than a single Gaussian, with this addressed by the proposed model.

Rather than focusing on the overall (Frobenius) accuracy in the inverse mapping $\boldsymbol{Y} \leftarrow \boldsymbol{X}$, there has also been an emphasis on designing embeddings $\boldsymbol{Y} \rightarrow \boldsymbol{X}$ for which the ℓ_2 distances between any two columns of $\boldsymbol{Y}$ is retained accurately when considering the corresponding ℓ_2 distance between the corresponding columns of $\mathbf{X}$ (preservation of inter-element distances). Motivated by the Johnson-Lindenstrauss Lemma [JL82, Ach03], and the corresponding proofs, random projections of various forms have been considered. Specifically, in this case $\boldsymbol{x}_i = \boldsymbol{\Phi}\boldsymbol{y}_i$, where the elements of $\boldsymbol{\Phi} \in \mathbb{R}^{d \times D}$ are constituted at random, based on one of several possible distributions [Ach03]. The mapping $\boldsymbol{Y} \leftarrow \boldsymbol{X}$ is under-determined, but if each $\boldsymbol{y}_i$ may be sparsely represented in some basis, the recovery $\boldsymbol{Y} \leftarrow \boldsymbol{X}$ may be achieved accurately, even *perfectly* in the noise-free case; this corresponds to compressive sensing (CS) [CT06]. This CS construction has been generalized by only requiring that the columns of $\boldsymbol{Y}$ come from a union of subspaces [EM09], which is often of interest in practice. In the case of CS, if one wants the opportunity to perform the inversion $\boldsymbol{Y} \leftarrow \boldsymbol{X}$, it is important to note that the required embedding dimension d is dependent on the sparseness with which the columns of $\boldsymbol{Y}$ may be rendered in an appropriate basis [CT06] (or based upon the properties of the union-of-subspace model from which $\boldsymbol{Y}$ are drawn); these characteristics of $\boldsymbol{Y}$ may not be known *a priori*.

While random-projection-based embeddings may be used to retain inter-element ℓ_2 distances, it has recently been recognized that this may not be the appropriate metric for data of interest. Specifically, in many real problems the columns of $\boldsymbol{Y}$ may be drawn from a manifold, and the ℓ_2 distance between $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ may be quite different from the corresponding geodesic distance along the manifold. This insight

has motivated embedding methods such as kernel PCA [SSM98], Isomap [TdSL00], local-linear embeddings (LLE) [RS00] and diffusion methods [CL06]. By exploiting the low-dimensional manifold of many realistic data sets, the embedding dimension d often may be made very small (linked to the intrinsic dimensionality of the subspace of $\mathbb{R}^D$ from which the columns of $\boldsymbol{Y}$ are drawn). However, these methods generally suffer from an inability to perform the inversion from the embedding space, $\boldsymbol{Y} \leftarrow \boldsymbol{X}$. One way this limitation may be overcome is by learning the embedding $\boldsymbol{Y} \rightarrow \boldsymbol{X}$ by one of these methods, and then retaining exemplar $\boldsymbol{y}_i$ for local regions of the embedding space; these exemplars or landmarks may be used to perform regression to approximately achieve $\boldsymbol{Y} \leftarrow \boldsymbol{X}$. A question arises concerning the number of such exemplars and their distribution; this chapter presents a method for addressing this question (rather than explicitly selecting particular landmarks from $\{\boldsymbol{y}_i\}_{i=1,N}$, we learn summary low-dimensional factor loadings, in a factor model).

The proposed method is based on a statistical analysis of the columns of $\boldsymbol{Y}$, moving beyond the simple Gaussian assumption. A *mixture* of factor analyzers (MFA) [GB00] may be used to model data that live on a manifold, with SVD essentially recovered in the case of a single mixture component. Previous related statistical approaches include probabilistic PCA [TB99b] and the more-recent Gaussian process latent variable model (GP-LVM) [Law05]. The PPCA model is limited by a simple Gaussian assumption, and a limitation of GP-LVM is that it is difficult to embed a new high-dimensional sample in the latent space, after the model has been learned. The proposed MFA method may be viewed as an extension of PPCA to a mixture of Gaussians, with a method presented to infer the proper number of mixture components, and the dimensionality or rank of each local Gaussian.

The direct utilization of an MFA poses challenges for learning an embedding based on the low-dimensional factor scores (*i.e.*, with $\boldsymbol{x}_i$ defined by the factor scores). Specifically, the MFA does not impose that if $\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2$ is small that the associated

$\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2$ is also small; *i.e.*, it does not achieve property (iv) concerning preservation of relative distances in the embedding space. This is because $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ may be close but may be represented by distinct Gaussian components. The latent factor scores in $\mathbb{R}^d$ need not (and generally do not) preserve proximity of data in $\mathbb{R}^D$. There has been previous research on development of techniques for aligning the latent features of an MFA [Ver06, ZZ04, TR03, VRV04]. The method in [Ver06] used a MAP solution, and is not directly transferable to the fully Bayesian solution of interest here (which allows us to infer a proper number of mixture components, and the dimensionality of each component). The approach in [ZZ04] is effective, but the inferred model is not explicitly statistical in nature (and again the latent-space dimensionality is set), and consequently it is not appropriate for the proposed model. There has been much research [TR03, VRV04] on development of two-step processes, in which one first learns an MFA, and then subsequently aligns it. However, two step approaches are well known to be sub-optimal in treating the results of the first step as fixed and known in implementing the second step. We therefore develop a new one-step alignment procedure that is compatible with hierarchical Bayesian formulations of the type developed here, in which the goals of MFA learning and latent-space alignment are addressed simultaneously. Further, the proposed method allows us to achieve all four embedding goals elucidated above.

To demonstrate the utility of such embeddings for analysis of high-dimensional data, we consider analysis of high-dimensional dynamic data, specifically motion-capture video [THR07, WFH08]. We perform analysis of the dynamic data, like considered in [FSJW10], with the goal of inferring inter-relationships between different portions of the time-dependent data. This is a general problem in the analysis of time-evolving data, with the motion-capture data presenting an example for which the results may be easily interpreted and assessed. Unlike [FSJW10] we also develop a model capable of accurate dynamic-motion *synthesis*, analogous to

[THR07, WFH08, LM07, GMHP04]. The dynamic-motion model is learned in the embedding space, and the ability to map from the embedding space to the original $\mathbb{R}^D$ space is performed using the properties of the developed statistical embedding method.

The methods employed to perform the above analysis are based upon tools from nonparametric Bayesian analysis. Specifically, within the context of the embedding, the number of mixture components in the MFA is inferred based on the data, using the Dirichlet process (DP) [Fer73], and the dimension of each individual mixture component (rank) is inferred via a variation of the Indian buffet process [GG05], implemented via a truncated beta-Bernoulli process [TJ07]. A new framework is developed for aligning the factor scores, to achieve a useful embedding (such that the concept of proximity in the space $\boldsymbol{X}$ is compatible with that in $\boldsymbol{Y}$). Concerning the dynamic model, we employ a separate DP-based mixture model in the low-dimensional space of factor scores, and within each mixture component a local, Markov linear-regression model is employed. The overall dynamic model is therefore nonlinear. All inference is performed via efficient Gibbs sampling.

The remainder of the chapter is organized as follows. In Section 3.2 we introduce a method for designing an MFA, while simultaneously aligning the factor scores. In Section 3.3 we describe the method developed for analyzing the dynamics of time-dependent data, with this analysis performed in the low-dimensional embedding space. We also discuss how this model may be used with an inverse embedding to synthesize dynamic data. In Section 3.4 we present several sets of results. Specifically, we examine the ability of the model to infer a proper dimensionality d of the data $\boldsymbol{Y}$, we demonstrate the ability of the model to embed new $\boldsymbol{y}_{N+1}$ in $\mathbb{R}^d$ without having to go back to the training data $\{\boldsymbol{y}_i\}_{i=1,N}$, we perform comparisons to other embedding procedures from the literature on "standard" embedding database, and finally we consider analysis and synthesis of dynamic motion-capture data.

3.2 Aligned Mixture of Factor Analyzers

3.2.1 Mixture of factor analyzers

A factor-analysis (FA) model may be represented as

$$\boldsymbol{y}_i = \boldsymbol{A}\boldsymbol{\Lambda}\boldsymbol{x}_i + \boldsymbol{\mu} + \boldsymbol{\epsilon}_i \tag{3.1}$$

where $\boldsymbol{A} \in \mathbb{R}^{D\times K}$ is the factor loading matrix, $\boldsymbol{\Lambda} \in \mathbb{R}^{K\times K}$ is a sparse diagonal matrix, $\boldsymbol{x}_i \in \mathbb{R}^K$, $\boldsymbol{\mu} \in \mathbb{R}^D$, and $\boldsymbol{\epsilon}_i \in \mathbb{R}^D$ represents noise or residual. The following priors are imposed:

$$\boldsymbol{a}_k \sim \mathcal{N}(\boldsymbol{0}, \frac{1}{D}\boldsymbol{I}_D), \quad \boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{0}, \frac{1}{K}\boldsymbol{I}_K), \quad \boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{0}, \gamma^{-1}\boldsymbol{I}_D), \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(0, \alpha_0^{-1}\boldsymbol{I}_D) \tag{3.2}$$

where $\boldsymbol{a}_k$ is the kth column of $\boldsymbol{A}$, $k = 1, \ldots, K$. Note that the expected ℓ_2 norm of each $\boldsymbol{a}_k$ is one. Additionally, gamma hyper-priors are employed for γ and α_0, with details on setting hyper-parameters discussed when presenting results.

Concerning the diagonal matrix $\boldsymbol{\Lambda}$, we assume $\boldsymbol{\Lambda} = \text{diag}(\lambda_1 b_1, \lambda_2 b_2, \ldots, \lambda_K b_K)$, where $(b_1, \ldots, b_K)$ is designed to be a sparse binary vector. Specifically, for $k \in \{1, \ldots, K\}$, we impose

$$\lambda_k \sim \mathcal{N}(0, 1/\beta), \quad b_k \sim \text{Bernoulli}(\pi_k), \quad \pi_k \sim \text{Beta}(a/K, b(K-1)/K) \tag{3.3}$$

with a gamma hyper-prior placed on β. The elements $(\lambda_1, \ldots, \lambda_K)$ play the role of singular values, although we do not explicitly impose that these are positive, with the sign absorbed into the factor loading.

The beta-Bernoulli construction of the binary vector $\boldsymbol{b} = (b_1, \ldots, b_K)$ is similar to the manner in which binary *matrices* are constructed via the Indian buffet process [ZCP+09, TJ07, GG05]. This understanding assists in analyzing the prior information the above construction imposes concerning the dimensionality d of the latent space. Specifically, by marginalizing out the probability vector $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)$, the

number of non-zero elements of $\boldsymbol{b}_k$ is drawn Binomial$(K, a/(a+b(K-1)))$, and as $K \rightarrow \infty$ the number of nonzero components is distributed Poisson(a/b). This explicit understanding of the statistics imposed for d within the prior is an attractive characteristic of the above construction, relative to hierarchical shrinkage priors like the Student-t [Tip01] (which do not explicitly impose sparseness).

Upon marginalizing out $\boldsymbol{x}_i$, the conditional density of $\boldsymbol{y}_i$ is

$$p(\boldsymbol{y}_i|\boldsymbol{A}, \boldsymbol{\mu}, \alpha_0, \beta) = \mathcal{N}(\boldsymbol{\mu}, \frac{1}{K}\boldsymbol{A}\boldsymbol{\Lambda}^2\boldsymbol{A}^T + \alpha_0^{-1}\boldsymbol{I}_D) \tag{3.4}$$

Each $\boldsymbol{y}_i$ is consequently assumed drawn from a *linear* subspace spanned by the columns of $\boldsymbol{A}\boldsymbol{\Lambda}$, and typically $\boldsymbol{A}\boldsymbol{\Lambda}^2\boldsymbol{A}^T$ is of low rank, since $\boldsymbol{\Lambda}$ has a sparse diagonal. Hence, $\boldsymbol{y}_i$ is drawn from a low-dimensional portion of $\mathbb{R}^D$, centered about $\boldsymbol{\mu}$. The FA model therefore affords a Bayesian form of SVD, with the columns of $\boldsymbol{A}\boldsymbol{\Lambda}$ essentially spanning the space of the principal singular vectors in an SVD analysis. Note that we do not explicitly impose that the columns of $\boldsymbol{A}$ are orthogonal, as they are in an SVD analysis. It is possible to impose that the factor loadings are orthogonal within the Bayesian analysis [Hof09], but this comes at significant computational expense, and we have found the above procedure to work well in practice.

A limitation of the above model is that it assumes all data are drawn from a single Gaussian and, as indicated, from an alternative viewpoint this implies that all $\boldsymbol{y}_i$ are drawn from the same linear subspace. A natural extension to a nonlinear model is achieved by assuming $\boldsymbol{y}_i$ is drawn from a *mixture* of FA (MFA) models [GB00]. A natural questions concerns the number of mixture components, with that addressed here by use of the Dirichlet process (DP) [Fer73]; one may also consider a more-general construction in terms of the Pitman-Yor process [IJ01], but that is not considered here. The DP-based MFA can be written as

$$\boldsymbol{y}_i = \boldsymbol{A}_{z(i)}\boldsymbol{\Lambda}_{z(i)}(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_{z(i)}) + \boldsymbol{\mu}_{z(i)} + \boldsymbol{\epsilon}_i \tag{3.5}$$

with

$$z(i) \sim \sum_{j=1}^{\infty} w_j \delta_j, \qquad w_j = V_j \prod_{l=1}^{j-1}(1 - V_l), \qquad V_j \sim \text{Beta}(1, \alpha) \tag{3.6}$$

with $\{\boldsymbol{A}_j\}_{j=1,\infty}$, $\{\boldsymbol{\mu}\}_{j=1,\infty}$ and $\{\boldsymbol{\Lambda}\}_{j=1,\infty}$ drawn as above (3.2). The expression $z(i)$ is an integer indicator variable, denoting which mixture component data $\boldsymbol{y}_i$ is associated with. In practice the number of "sticks" w_j is truncated [IJ01], and when truncating to J sticks, $V_J = 1$, and $z(i) \sim \sum_{j=1}^{J} w_j \delta_j$. Concerning $\{\boldsymbol{\Lambda}\}_{j=1,J}$, within the DP-based construction we draw a distinct $\boldsymbol{\pi}_j = (\pi_{j1}, \ldots, \pi_{jK})$ for each of the J mixture components, from which the respective mixture-dependent binary vector $\boldsymbol{b}_j = (b_{j1}, \ldots, b_{jK})$ is drawn. Note that in general the number of non-zero components in the diagonal of $\boldsymbol{\Lambda}_j$ varies with j, which implies that the data dimensionality may vary across the support of the data (*i.e.*, the rank of the individual mixture components is not constrained to be the same).

The remaining components of the model are drawn

$$\hat{\boldsymbol{\mu}}_j \sim \mathcal{N}(0, \xi^{-1}\boldsymbol{I}_d), \qquad \boldsymbol{x}_i \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_{z(i)}, \beta_{z(i)}^{-1}\boldsymbol{I}_d) \tag{3.7}$$

with $\boldsymbol{\epsilon}_i$ drawn as in (3.2). Gamma hyperpriors are placed on α, ξ and $\{\beta_j\}_{j=1,J}$. Note that the introduction of $\hat{\boldsymbol{\mu}}_j$ is nonstandard in MFA modeling. The idea is to model *offsets* for local clusters in latent space, in preparation for subsequent alignment. This is detailed in the next subsection.

The procedure in [GB00] may also be used to infer the number of mixture components and their rank, analogous to the method discussed above. In that analysis shrinkage priors are used to infer the latent-space dimensionality, rather than the spike-slab prior discussed above, and the Dirichlet distribution was employed to infer the number of mixture components. However, the model in [GB00] does not align the latent space, and when shrinkage priors of the form in [GB00] were employed within the one-step alignment discussed below, the performance was inferior to that

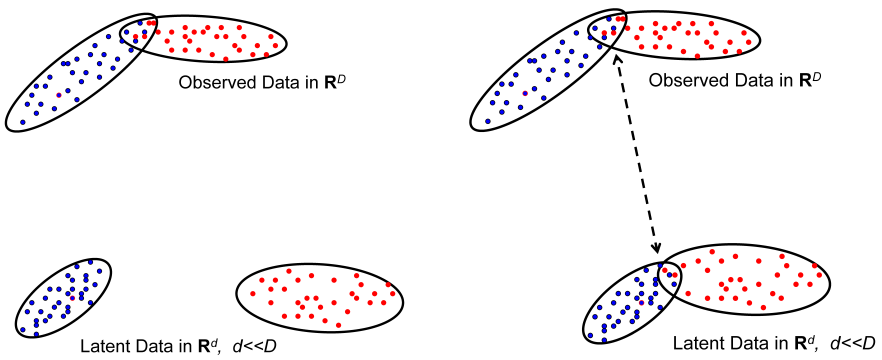


FIGURE 3.1: Schematic of two Gaussians in a mixture model. In the original high-dimensional space $\mathbb{R}^D$ the mixtures partially overlap (top), while this is not guaranteed in the low-dimensional latent space $\mathbb{R}^d$, $d \ll D$. At bottom-right, the high-dimensional mixture components and the latent $\boldsymbol{x}_i$ are aligned, in that if $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ are proximate, so are the corresponding latent $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$.

of the spike-slab prior, motivating our use of the latter.

3.2.2 One-step MFA alignment

To appreciate the limitations of directly using the model in (3.5) to perform an embedding, consider Figure 3.1. At top are the $\{\boldsymbol{y}_i\}_{i=1,N}$, in Figure 3.1 represented in the form of two partially overlapping Gaussians; the colors indicate which of the two Gaussians the data are associated with. For this simple two-Gaussian example, considering (3.5), the Gaussians in the high-dimensional space $\mathbb{R}^D$ are centered at $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$; in the latent space the mixtures for $\{\boldsymbol{x}_i\}_{i=1,N}$ in $\mathbb{R}^d$ are centered correspondingly at $\hat{\boldsymbol{\mu}}_1$ and $\hat{\boldsymbol{\mu}}_2$. As constituted thus far, there are no constraints within the model on $\hat{\boldsymbol{\mu}}_1$ and $\hat{\boldsymbol{\mu}}_2$ that impose the alignment at the bottom-right in Figure 3.1. This implies that if $\boldsymbol{y}_i$ and $\boldsymbol{y}_j$ are from distinct Gaussians, but with small $\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2$, there are no assurances that $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2$ will also be small; consequently, the $\boldsymbol{x}_i$ do not serve as effective low-dimensional features of $\boldsymbol{y}_i$.

Thus far we have not explicitly utilized knowledge of inter-sample distances $\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2$, with the exploitation of such distances at the heart of many previously developed embedding strategies [TdSL00, RS00, CL06]. With the goal of

achieving a related construction, consider a general kernel $\mathcal{K}(\boldsymbol{y}_i, \boldsymbol{y}_j; \Theta)$, where Θ are kernel-dependent parameters, and $\mathcal{K}(\boldsymbol{y}_i, \boldsymbol{y}_i; \Theta) = 1$ with $\mathcal{K}(\boldsymbol{y}_i, \boldsymbol{y}_j; \Theta) = 0$ in the limit $\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2 \rightarrow \infty$ (and for all i and j, $\mathcal{K}(\boldsymbol{y}_i, \boldsymbol{y}_j; \Theta) \leq 1$). For example, in our analysis we employ a radial basis function (RBF)

$$\mathcal{K}(\boldsymbol{y}_i, \boldsymbol{y}_j; \sigma^2) = \exp[-\frac{1}{\sigma^2}\|\boldsymbol{y}_i - \boldsymbol{y}_j\|_2^2] \tag{3.8}$$

As in diffusion analysis [CL06], we define a random-walk matrix for the probability of "walking" from $\boldsymbol{y}_i$ to $\boldsymbol{y}_j$ in a single step,

$$W(i,j) = \mathcal{K}(\boldsymbol{y}_i, \boldsymbol{y}_j; \sigma_i^2) / \sum_{k=1}^{N} \mathcal{K}(\boldsymbol{y}_i, \boldsymbol{y}_k; \sigma_i^2) \tag{3.9}$$

We employ a distinct σ_i for each $\boldsymbol{y}_i$, as in [ZMP04], such that with high probability we may only walk to a prescribed number of nearest neighbors of $\boldsymbol{y}_i$. The key to most of these embeddings, as well as many semi-supervised-learning models [KWX+04] is that $\boldsymbol{y}_i \approx \sum_{j=1}^{N} W(i,j)\boldsymbol{y}_j$, particularly if the RBF kernel parameter σ_i is constructed such that $W(i,j)$ only has appreciable amplitude for $\boldsymbol{y}_j$ within a close neighborhood of $\boldsymbol{y}_i$. Hence, $\boldsymbol{y}_i$ is approximately equal to a weighted average of $\boldsymbol{y}_j$ within a nearby neighborhood. The simple modification to the MFA involves imposing this same structure defined by the $W(i,j)$ on the latent $\{\boldsymbol{x}_i\}_{i=1,N}$, and the centers of the mixture components in the latent space, $\{\hat{\boldsymbol{\mu}}_j\}_{j=1,N}$ are therefore adjusted to achieve this end.

Assume that the mixture means in the latent space $\{\hat{\boldsymbol{\mu}}_j\}_{j=1,J}$ are drawn as specified above, as is the truncated stick-breaking construction $\sum_{j=1}^{J} w_j \delta_j$. We now consider the following refined hierarchical construction for $\boldsymbol{x}_i$:

$$\boldsymbol{x}_i = \sum_{k=1}^{N} W(i,k)\boldsymbol{x}_k^*, \qquad \boldsymbol{x}_k^* \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_{z(k)}, \beta_{z(k)}^{-1}\boldsymbol{I}_K), \qquad z(k) \sim \sum_{j=1}^{J} w_j \delta_j \tag{3.10}$$

The set of latent vectors $\{\boldsymbol{x}_k^*\}_{k=1,N}$ are drawn as before, but they are not directly used as the latent features for the corresponding data. Rather, the $\boldsymbol{x}_i$ associated with $\boldsymbol{y}_i$ is represented as a linear combination of the $\boldsymbol{x}_k^*$ that are in a neighborhood of sample $\boldsymbol{y}_i$, with the idea of a neighborhood defined by the original high-dimensional data $\{\boldsymbol{y}_i\}_{i=1,N}$. This implies a similarity among all $\{\boldsymbol{x}_j\}_{j\in\mathcal{N}_i}$ that are within a neighborhood $\mathcal{N}_i$ of $\boldsymbol{x}_i$; the concept of a neighborhood is defined by $W(i,j)$, where proximity in $W(i,j)$ is constituted by the associated high-dimensional data $\boldsymbol{y}_i$.

Let $\mathcal{N}_\epsilon^{\boldsymbol{y}}(i) = \{\boldsymbol{y} : ||\boldsymbol{y} - \boldsymbol{y}_i||_2^2 < \epsilon\}$ denote an ϵ-sized L_2 neighborhood around $\boldsymbol{y}_i$. Then, for small ϵ, it is appealing for $\boldsymbol{x}_k$ to be close to $\boldsymbol{x}_i$ for all k such that $\boldsymbol{y}_k \in \mathcal{N}_\epsilon^{\boldsymbol{y}}(i)$. In particular, we would like ℓ_2 neighborhoods to be maintained in mapping from $\boldsymbol{Y} \to \boldsymbol{X}$, so $\boldsymbol{y}_k \in \mathcal{N}_\epsilon^{\boldsymbol{y}}(i)$ implies $\boldsymbol{x}_k \in \mathcal{N}_{\epsilon^*}^{\boldsymbol{x}}(i)$, with $\mathcal{N}_{\epsilon^*}^{\boldsymbol{x}}(i)$ an ϵ^*-sized L_2 neighborhood around $\boldsymbol{x}_i$ and ϵ^* a small positive constant that decreases with ϵ; this is achieved by the above construction.

Note that we only use the random-walk matrix $\boldsymbol{W}$ to learn the aligned MFA based on training data $\boldsymbol{Y}$. Once the model is so learned, the mapping $\boldsymbol{y}_{N+1} \to \boldsymbol{x}_{N+1}$ is performed using the learned model (discussed in Section 3.2.5), and $\boldsymbol{W}$ is not updated and the training data $\boldsymbol{Y}$ are not needed. This is an important distinction with techniques like diffusion [CL06], which must augment the random-walk matrix, using $\boldsymbol{Y}$, to embed a new $\boldsymbol{y}_{N+1}$.

3.2.3 Nonlinear spectral regression & MFA alignment

The form of the model in (3.10), particularly the relationship $\boldsymbol{x}_i = \sum_{k=1}^N W(i,k)\boldsymbol{x}_k^*$, is suggestive of an alternative alignment procedure, manifesting a statistical form of diffusion analysis [CL06]. Specifically, recall the relationship

$$\boldsymbol{W}\boldsymbol{\psi}_j = \hat{\lambda}_j \boldsymbol{\psi}_j \tag{3.11}$$

where $\hat{\lambda}_j$ is the jth eigenvalue and $\boldsymbol{\psi}_j \in \mathbb{R}^N$ the jth eigenvector of the random-walk matrix $\boldsymbol{W}$. The kth row of the $N \times N$ matrix $(\hat{\lambda}_2\boldsymbol{\psi}_2, \ldots, \hat{\lambda}_N\boldsymbol{\psi}_N)$ defines the diffusion coordinates/embedding of $\boldsymbol{y}_k$ [CL06]. The eigenvalues are assumed arranged $\hat{\lambda}_1 = 1 \geq \hat{\lambda}_2 \geq \cdots \geq \hat{\lambda}_N$, and typically only a small fraction of the N eigenvalues have significant values. This and (3.10) suggests the following alternative MFA model:

$$\boldsymbol{y}_i = \boldsymbol{A}_{z(i)}\hat{\boldsymbol{\Lambda}}_{z(i)}(\hat{\boldsymbol{\psi}}_i - \hat{\boldsymbol{\mu}}_{z(i)}) + \boldsymbol{\mu}_{z(i)} + \boldsymbol{\epsilon}_i \tag{3.12}$$

where $\hat{\boldsymbol{\Lambda}}_j = \text{diag}(\hat{\lambda}_2 b_{j1}, \ldots, \hat{\lambda}_{K+1} b_{jK})$ and $\hat{\boldsymbol{\psi}}_i = (\psi_{i2}, \ldots, \psi_{i(K+1)})^T$, assuming $K+1 \leq N$, where ψ_{ik} is the ith component of the kth diffusion eigenvector $\boldsymbol{\psi}_k$; the first coordinate ψ_{i1} is a constant for all samples, and is effectively accounted for by the mean $\boldsymbol{\mu}_{z(i)}$. The observed $\hat{\boldsymbol{\psi}}_i$, which like $\boldsymbol{y}_i$ is associated with cluster $z(i)$, is assumed drawn from a corresponding cluster in the latent space

$$\hat{\boldsymbol{\psi}}_i \sim \mathcal{N}(\hat{\boldsymbol{\mu}}_{z(i)}, \beta_{z(i)}^{-1}\mathbf{I}_K) \tag{3.13}$$

with a gamma prior placed on the $\beta_{z(i)}$, analogous to how $\boldsymbol{x}_k^*$ was drawn in (3.10). Hence, this model performs *joint* learning of mixture models in the space of $\{\boldsymbol{y}_i\}_{i=1,N}$ and $\{\hat{\boldsymbol{\psi}}_i\}_{i=1,N}$, with the two mixture components linked via the corresponding, mixture-component-dependent factor loadings $\boldsymbol{A}_{z(i)}$. Note that because of the aforementioned properties of the eigenvalues and eigenvectors of $\boldsymbol{W}$, the desired relationship $\boldsymbol{x}_i = \sum_{k=1}^N W(i,k)\boldsymbol{x}_k^*$ in the low-dimensional space is satisfied automatically, where $\boldsymbol{x}_k^* = \hat{\boldsymbol{\psi}}_k$ and the sparse vector $\boldsymbol{b}_j$ selects the principal eigenvalues. Further, the model inherits other properties of diffusion analysis, namely that in the embedding space Euclidian distances corresponded to diffusion distances in the high-dimensional space [CL06] (thereby capturing geodesic properties of the support of the data).

Examining (3.12), another way to view this model is as regression between the $\boldsymbol{y}_i$ and diffusion coordinates $(\hat{\lambda}_2\psi_{i2}, \ldots, \hat{\lambda}_{K+1}\psi_{i(K+1)})$. The factor loadings $\boldsymbol{A}_{z(i)}$ constitute the regression coefficients, and with $\hat{\boldsymbol{\mu}}_{z(i)}$ and $\boldsymbol{\mu}_{z(i)}$ centering each of the mixture

components in the two domains. In this setting we perform *joint* mixture modeling in the space of the original high-dimensional data $\boldsymbol{y}_i$ and in the embedding coordinates $(\hat{\lambda}_2\psi_{i2}, \ldots, \hat{\lambda}_{K+1}\psi_{i(K+1)})$, with the two mixtures coupled via the factor loadings $\boldsymbol{A}_{z(i)}$. We therefore refer to this as nonlinear spectral regression (NSR).

In our original construction in Section 3.2.1 the columns of the matrices $\boldsymbol{A}_z$ were constrained to have unit expected norm, as they were drawn from $\mathcal{N}(\boldsymbol{0}, \frac{1}{N}\boldsymbol{I}_D)$. In the NSR formulation of this section, the factor score associated with $\boldsymbol{y}_i$ is defined by the K-dimensional diffusion coordinates, and now the columns of $\boldsymbol{A}_z$ (which perform the regression) are drawn from $\mathcal{N}(\boldsymbol{0}, \zeta^{-1}\boldsymbol{I}_D)$, with ζ drawn from a gamma prior (removing the expected unit-norm constraint on the factor loadings). Hence, now, rather than strongly constraining the factor loadings we constrain the factor scores, and associate them with spectral coordinates [CL06] (since the model depends on the product of the factor scores and loadings, the strong constraints may be interchanged). The rest of the model remains unchanged, and the *nonlinear* regression of the spectral coordinates is manifested because each mixture component in the MFA is a *local* linear relationship between the spectral coordinates and associated high-dimensional data, but the multiple mixture components cumulatively constitute a nonlinear regression model.

While we have been motivated by diffusion analysis [CL06], one may similarly use other spectral methods for representation of $\boldsymbol{x}_i$, and results are presented below based on both diffusion maps and Isomap [TdSL00] (these representing special cases of *spectral* methods [HLMS04]). The model is *locally* linear within any particular mixture component, but the mapping of the data to a particular mixture component yields an algorithm that is overall nonlinear.

In our experiments, both the alignment method in (3.10) and NSR have worked equally well in the context of modeling and analysis of dynamic high-dimensional data (discussed in the next section). If one is only interested in an embedding, without

dynamic analysis, we recommend the NSR formulation of this section, because of its close linkage to spectral technology, which has nice theoretical properties [CL06, HLMS04]. Further, as discussed when presenting results, the NSR formulation has been found to provide an accurate estimate of the latent-space dimension. This is performed by inferring the number of non-zero components in the binary vectors $(b_{j1}, \ldots, b_{jK})$. Computation of an approximate posterior distribution on all model parameters is performed via Gibbs sampling, as summarized in the Appendix B.

3.2.4 Reversing embeddings

Assume that we are given an embedded vector $\boldsymbol{x} \in \mathbb{R}^d$, where $d \leq K$ is the inferred dimensionality of the latent space. We wish to infer the associated data $\boldsymbol{y} \in \mathbb{R}^D$ in the original high-dimensional space ($D \gg d$). Based upon the model learned as discussed above, on the training data $\mathcal{D} = \{\boldsymbol{y}_i\}_{i=1,N}$, we infer a statistical distribution for the latent variable $\boldsymbol{x}$:

$$p(\boldsymbol{x}|\mathcal{D}) = \sum_{m=1}^{M} a_m \mathcal{N}(\boldsymbol{x}; \hat{\boldsymbol{\mu}}_m, \boldsymbol{\Sigma}_m) \tag{3.14}$$

where we approximate each mixture as a Gaussian (based upon the mean $\hat{\boldsymbol{\mu}}_m$ and covariance matrix $\boldsymbol{\Sigma}_m$ inferred from the Gibbs collection samples for mixture component m); the probability of each mixture component $a_m \geq 0$, $\sum_{m=1}^{M} a_m = 1$ is also here the average inferred from the Gibbs collection samples, with the number of mixture components M inferred from the data, based upon the truncated Dirichlet process. The *expected* mapping $\boldsymbol{x} \rightarrow \boldsymbol{y}$ is

$$\boldsymbol{y} = \sum_{m=1}^{M} \frac{a_m \mathcal{N}(\boldsymbol{x}; \hat{\boldsymbol{\mu}}_m, \boldsymbol{\Sigma}_m)}{\sum_{j=1}^{M} a_j \mathcal{N}(\boldsymbol{x}; \hat{\boldsymbol{\mu}}_j, \boldsymbol{\Sigma}_j)} [\boldsymbol{A}_m \boldsymbol{\Lambda}_m (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_m) + \boldsymbol{\mu}_m] \tag{3.15}$$

corresponding to the mapping from each mixture component alone, weighted by the probability that $\boldsymbol{x}$ (and hence $\boldsymbol{y}$) is associated with that mixture component. When

performing this mapping, we also employ mean values (from the Gibbs collection samples) for $\{\boldsymbol{A}_m, \boldsymbol{\Lambda}_m, \boldsymbol{\mu}_m\}_{m=1,M}$.

3.2.5 Statistical embeddings of new data

Based upon the training data $\mathcal{D}$, again assume we have learned the latent-space mixture model in (3.14), and assume that we have (mean) values for the corresponding mixture-component parameters. The generative process corresponds to first drawing a mixture component (shared between the low- and high-dimensional spaces), then drawing a latent vector $\boldsymbol{x}$, and finally mapping this vector to the high-dimensional space $\boldsymbol{y}$ via the mixture-component-dependent regression mapping (factor loading). If the components of $\boldsymbol{\epsilon}_i$ are zero-mean with precision α_0 (mean value again obtained from the Gibbs collection samples), then the *joint* probability of $\boldsymbol{y}$ and $\boldsymbol{x}$ is

$$p(\boldsymbol{y}, \boldsymbol{x}) = \sum_{m=1}^{M} a_m \mathcal{N}(\boldsymbol{A}_m \boldsymbol{\Lambda}_m (\boldsymbol{x} - \hat{\boldsymbol{\mu}}_m) + \boldsymbol{\mu}_m, \alpha_0^{-1} \mathbf{I}_D) \mathcal{N}(\boldsymbol{x}; \hat{\boldsymbol{\mu}}_m, \boldsymbol{\Sigma}_m) \tag{3.16}$$

where we have marginalized (summed) out the latent mixture component m. Because of conjugacy, one may analytically express

$$\begin{aligned} p(\boldsymbol{x}|\boldsymbol{y}) &= p(\boldsymbol{x}, \boldsymbol{y}) / \int p(\boldsymbol{x}, \boldsymbol{y}) d\boldsymbol{x} \\ &= \sum_{m=1}^{M} \tilde{a}_m \mathcal{N}(\boldsymbol{x}; \tilde{\boldsymbol{\mu}}_m, \tilde{\boldsymbol{\Sigma}}_m) \end{aligned} \tag{3.17}$$

with

$$\begin{aligned} \tilde{a}_m &= \frac{a_m \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_m, \alpha_0^{-1} \mathbf{I}_D + \boldsymbol{A}_m \boldsymbol{\Lambda}_m \boldsymbol{\Sigma}_m \boldsymbol{\Lambda}_m \boldsymbol{A}_m^T)}{\sum_{m=1}^{M} a_m \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_m, \alpha_0^{-1} \mathbf{I}_D + \boldsymbol{A}_m \boldsymbol{\Lambda}_m \boldsymbol{\Sigma}_m \boldsymbol{\Lambda}_m \boldsymbol{A}_m^T)} \\ \tilde{\boldsymbol{\Sigma}}_m &= (\boldsymbol{\Sigma}_m^{-1} + \alpha_0 \boldsymbol{\Lambda}_m \boldsymbol{A}_m^T \boldsymbol{A}_m \boldsymbol{\Lambda}_m)^{-1} \\ \tilde{\boldsymbol{\mu}}_m &= \tilde{\boldsymbol{\Sigma}}_m [\alpha_0 \boldsymbol{\Lambda}_m \boldsymbol{A}_m^T (y - \boldsymbol{\mu}_m + \boldsymbol{A}_m \boldsymbol{\Lambda}_m \hat{\boldsymbol{\mu}}_m) + \boldsymbol{\Sigma}_m^{-1} \hat{\boldsymbol{\mu}}_m] \end{aligned}$$

In the above computations, the following identity for normal distribution is used: $\mathcal{N}(\boldsymbol{y}; \boldsymbol{A}_m\boldsymbol{\Lambda}_m(\boldsymbol{x}-\hat{\boldsymbol{\mu}}_m)+\boldsymbol{\mu}_m, \alpha_0^{-1}\mathbf{I}_D)\mathcal{N}(\boldsymbol{x}; \hat{\boldsymbol{\mu}}_m, \boldsymbol{\Sigma}_m) = \mathcal{N}(\boldsymbol{x}; \tilde{\boldsymbol{\mu}}_m, \tilde{\boldsymbol{\Sigma}}_m)\mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_m, \alpha_0^{-1}\mathbf{I}_D + \boldsymbol{A}_m\boldsymbol{\Lambda}_m\boldsymbol{\Sigma}_m\boldsymbol{\Lambda}_m\boldsymbol{A}_m^T)$.

When performing the embedding $\boldsymbol{y} \rightarrow \boldsymbol{x}$ for new $\boldsymbol{y} \notin \mathcal{D}$, instead of yielding a point estimation for $\boldsymbol{x}$, we achieve the distribution $p(\boldsymbol{x}|\boldsymbol{y})$ from (3.17). When presenting the results, we use the mean value $\hat{\boldsymbol{\mu}}$ of the inferred $\boldsymbol{x}$, where $\hat{\boldsymbol{\mu}} = \sum_{m=1}^{M} a_m\tilde{\boldsymbol{\mu}}_m$.

Summarizing, the NSR method constitutes a new means of summarizing the statistical properties of spectral embedding methods like Isomap, LLE and diffusion analysis. By jointly inferring mixture models in the original high-dimensional and embedded low-dimensional spaces, with factor loadings constituting the locally linear regression, we manifest new capabilities concerning embedding new data and performing inverse embeddings. The mixtures effectively constitute learned landmarks in the data, between the high- and low-dimensional spaces, with the number of such inferred based upon the data. As discussed when presenting results, the number of non-zero components in the diagonal of $\hat{\boldsymbol{\Lambda}}_{z(i)}$ has also proven to provide an accurate estimate for the latent dimension d, which may vary across the data in a mixture-component-dependent manner.

3.3 Modeling Latent-Space Dynamics

The hierarchical constructions summarized above yield MFA models of high-dimensional data, with aligned latent space $\boldsymbol{X}$. This therefore offers the opportunity to model the dynamics of *time-evolving* high-dimensional data $\boldsymbol{y}_{t_1}, \boldsymbol{y}_{t_2}, \ldots, \boldsymbol{y}_{t_T}$, with the dynamic modeling performed in the associated low-dimensional latent space $\boldsymbol{x}_{t_1}, \boldsymbol{x}_{t_2}, \ldots, \boldsymbol{x}_{t_T}$. Once such a dynamic model is so learned, one may synthesize time-evolving data in the low-dimensional latent space and then map it to the high-dimensional space via the method in Section 3.2.4. Related objectives have been considered via GP-LVM for motion-capture data [WFH08, LM07, GMHP04].

3.3.1 Single-layer dynamic model

Assume we have the history of latent features $\boldsymbol{x}_{t_1}, \boldsymbol{x}_{t_2}, \ldots, \boldsymbol{x}_{t_T}$, which we wish to use to learn a nonlinear dynamic model. We build a Markovian model for $\boldsymbol{x}_t \rightarrow \boldsymbol{x}_{t+1}$, dividing the latent space into a set of Gaussian regions/clusters (generally distinct from those used in NSR), and there is a separate Markovian model for each such region. For any given $\boldsymbol{x}_t$, the new $\boldsymbol{x}_{t+1}$ is a linear combination of predictions based upon all mixture components, weighted by the probability that $\boldsymbol{x}_t$ is within the respective mixture regions.

Specifically, consider the following hierarchical model:

$$\boldsymbol{x}_{t+1} = \boldsymbol{M}_{z(t)}\boldsymbol{x}_t + \boldsymbol{\epsilon}_{t+1}^{dyn} , \qquad \boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{m}_{z(t)}, \boldsymbol{\Sigma}_{z(t)}^{-1}) , \qquad z(t) \sim \sum_{j=1}^{J^{dyn}} c_j \delta_j \quad (3.18)$$

where J^{dyn} is the number of sticks taken in a truncated-DP construction for the dynamic model, and $\boldsymbol{\epsilon}_{t+1}^{dyn} \sim \mathcal{N}(0, \gamma_0^{-1}\boldsymbol{I}_d)$ represent associated noise/error. Each component of each $\boldsymbol{M}_j$ is drawn i.i.d. from a zero-mean Gaussian distribution with diagonal precision matrix and gamma hyperprior, and each $(\boldsymbol{m}_j, \boldsymbol{\Sigma}_j)$ pair is drawn from a Gauss-Wishart prior. A gamma prior is again employed on the innovation parameter associated with the truncated stick-breaking construction.

The mapping $\boldsymbol{x}_t \rightarrow \boldsymbol{x}_{t+1}$ has a nonlinearity manifested because any prediction is a weighted sum of predictions from each of the J^{dyn} mixture components, with the relative degree to which the J^{dyn} mixture components contribute a function of $\boldsymbol{x}_t$. Specifically, we have

$$p(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t) = \sum_{j=1}^{J^{dyn}} w_j(\boldsymbol{x}_t)\mathcal{N}(\boldsymbol{M}_j\boldsymbol{x}_t, \gamma_0^{-1}\boldsymbol{I}_d) \quad (3.19)$$

with $w_j(\boldsymbol{x}_t) = c_j\mathcal{N}(\boldsymbol{x}_t|\boldsymbol{m}_j, \boldsymbol{\Sigma}_j^{-1})/\sum_{j'=1}^{J^{dyn}} c_{j'}\mathcal{N}(\boldsymbol{x}_t|\boldsymbol{m}_{j'}, \boldsymbol{\Sigma}_{j'}^{-1})$. Additionally, the Gibbs sampler used to learn this dynamic model of course also considers an ensemble of

model parameters, yielding further smoothing.

3.3.2 Hierarchical multi-task dynamic model

Consider the situation for which we have multiple distinct forms of dynamic data (*e.g.*, motion-capture data for people walking, running, dancing, etc.). Rather than learning embeddings and dynamic models for each data source in isolation, it is desirable to use all available data to learn the embedding and to jointly infer dynamic models. Such a procedure allows one to infer relationships between different types of data (*e.g.*, what motions in dance are similar to related motions in running). All available data may be aggregated when inferring statistical embeddings of the form discussion in Sections 3.2.2 and 3.2.3. To jointly model and analyze the dynamics of L different sources of dynamic data, we employ a *hierarchical* truncated-DP construction.

Let $\boldsymbol{x}_t^{(l)}$ represent the embedded data at time t, for data class $l \in \{1, \ldots, L\}$. The hierarchical model is

$$\boldsymbol{x}_{t+1}^{(l)} = \boldsymbol{M}_{l,t}\boldsymbol{x}_t^{(l)} + \boldsymbol{\epsilon}_{t+1,l}^{dyn}\,, \qquad \boldsymbol{x}_t^{(l)} \sim \mathcal{N}(\boldsymbol{m}_{l,t}, \boldsymbol{\Sigma}_{l,t}^{-1})\,, \qquad (\boldsymbol{M}_{l,t}, \boldsymbol{m}_{l,t}, \boldsymbol{\Sigma}_{l,t}) \sim G_l \quad (3.20)$$

where $G_l \sim \mathrm{DP}(\eta_2, GG_{02})$, and $G \sim \mathrm{DP}(\eta_1, G_{01})$. Recall from (3.6) that a draw G from a DP corresponds to a mixture of discrete atoms; the discrete atoms from G define different possible linear regressions, shared across the L different dynamic data sets under analysis. The continuous measure G_{02} defines task-dependent clustering of latent-feature space, extending (3.18) to the joint analysis of data from L forms of motion (multi-task learning). The measure G_{01} represents the prior on the regression matrix $\boldsymbol{M}$ (as above for analysis of a single form of motion), while G_{02} represents the Gauss-Wishart prior on $(\boldsymbol{m}, \boldsymbol{\Sigma})$. Gamma priors are again placed on η_1 and η_2. This yields a generalized hierarchical DP construction [TJBB06] for the local linear-regression parameters. Truncated stick-breaking representations [IJ01] are employed

for all DP implementations. The Gibbs update equations for the dynamic model are summarized in the Appendix B.

3.4 Experiments

3.4.1 Hyperparameter settings

The model may appear to have many hyperparameters, but in reality these parameters are set in a "standard" way [Tip01], with no tuning. Specifically, for all embedding experiments in this chapter, the hyperparameter values are $a_0 = 0.02$, $b_0 = 1$, $\xi = 1$ and all gamma priors are set to Gamma$(10^{-6}, 10^{-6})$. The hyperparameter values for the dynamic model are as follows: $u_0 = 0.1$, $\nu_0 = K + 2$, and $\boldsymbol{m}_0$ and $\boldsymbol{B}_0$ are set to the sample mean and sample precision of the training set, respectively. All gamma priors are equal to Gamma$(10^{-6}, 10^{-6})$, as in the embedding experiments. The model was not found to be sensitive to deviations from these standard settings, hence no attempt was made to optimize the hyperparameters.

3.4.2 Latent-space dimensionality estimation

The proposed nonlinear spectral regression (NSR) model provides an estimate of the latent-space dimension, by counting the number of nonzero elements of the diagonal matrices $\boldsymbol{\Lambda}_z$. The Gibbs sampler provides an estimate of the full posterior distribution on this number, and one example of such a posterior distribution is presented below. In the following experiments on dimensionality estimation we present the posterior mean as our estimate. When implementing NSR dimensionality estimation, we first analyze the data using Isomap [TdSL00], and retain the K most significant features, and use these within the method discussed in Section 3.2.3; the model infers which subset of the K spectral features are needed to represent the data $\boldsymbol{y}_i$, via the factor loadings, with the remaining spectral features discarded and $\boldsymbol{\epsilon}_i$ accounting for the residual. In these experiments we employed truncation $K = 20$, and we truncate

the number of mixture components to $J = 20$. A total of 500 burn-in Gibbs samples were employed, with 50 collection samples.

We compare the NSR dimensionality estimates with those from the following methods: Maximum Likelihood Estimation (MLE), Eigenvalue thresholding, Geodesic Minimum Spanning Tree (GMST) and Correlation Dimension (CorrDim). These methods are all described in [LB04] and references therein.

We have experimented with simulated spheres, balls, cubes and Gaussians of dimensions ranging from two to 25, embedded in 100-dimensional observation space. For brevity, we only show results in Figure 3.2 for a 6-dimensional ball, a 4-dimensional cube, Gaussians of dimension 8 and 10, a 2-dimensional Swiss roll, and a 2-dimensional torus, the two latter datasets being embedded in three-dimensional observation space; the Swiss-roll example is from [TdSL00]. We consider additive zero-mean, i.i.d. Gaussian noise with standard deviations σ_{noise} set to 0, 0.01, 0.05 and 0.1. For these examples, these noise levels correspond to the signal-to-noise (SNR) ratios presented in Table 3.1, which are computed according to $\text{SNR} = 10\log_{10}(\frac{\|\boldsymbol{x}\|^2_{\text{av}}}{D\sigma^2_{\text{noise}}})$, with $\|\boldsymbol{x}\|^2_{\text{av}}$ being the average squared ℓ_2 norm of the vectors drawn from each manifold.

Table 3.1: Signal-to-noise (SNR) ratios for the dimensionality estimation experiments.

Noise levels	0	0.01	0.05	0.1
6D ball	∞	19 dB	5 dB	-1 dB
4D cube	∞	21 dB	7 dB	1 dB
8D/10D Gaussian	∞	29/30 dB	15/16 dB	9/10 dB
2D torus	∞	31 dB	17 dB	11 dB
2D Swiss-roll	∞	34 dB	20 dB	14 dB

The results are averaged across 100 realizations of the noise and are representative of the wider set of experiments, with NSR yielding the best estimates. Note that NSR consistently provides the best dimensionality estimation, particularly at high noise levels.

Note that the NSR method provides highly accurate estimates of the dimensionality of the data, up to a noise standard deviation of 0.1 (SNR as low as -1 dB). To get a sense of when such dimensionality estimation breaks down with increasing noise level, we reconsider the Swiss-roll data, which is readily visualized since it is in three dimensions. In Figure 3.3 we depict the noise-free data (from [TdSL00]), and also show example data draws for noise standard deviation 0.1 and 0.2. All methods, including NSR, fail for $\sigma_{\text{noise}} = 0.2$ (SNR 8 dB). For this noise level, the surfaces of the roll come together, and therefore it is not surprising that the model no longer estimates the data dimension to be two.

3.4.3 Embeddings

To further demonstrate embedding performance, we consider the following widely studied datasets: the teapot data [TR03], the MNIST digit database, and two face datasets from [TdSL00]. We again use Isomap to supply the latent coordinates for the NSR formulation. The above results indicate that NSR infers the latent-space (embedding) dimensionality effectively, and therefore an important aspect of this subsection concerns the efficient embedding of new data without having to return to the training data, and also the ability to synthesize new data. In these examples the truncation level of the latent space is $K = 30$, and the truncation level on the number of mixture components is $J = 60$. We employed 2000 burn-in Gibbs samples, and 500 collection samples.

The teapot dataset is comprised of rotated teapot images. There are 400 RGB images, each one of size 101×76. Figure 3.4 shows the NSR embedding, which is smooth, and has no self-intersections and exhibits the expected circular topology. Also for the teapot data, we compare the out-of-sample performance of NSR to that of Isomap with the Nyström approximation. Note that, unlike traditional spectral methods, NSR does *not* require the Nyström approximation [DM05] to embed

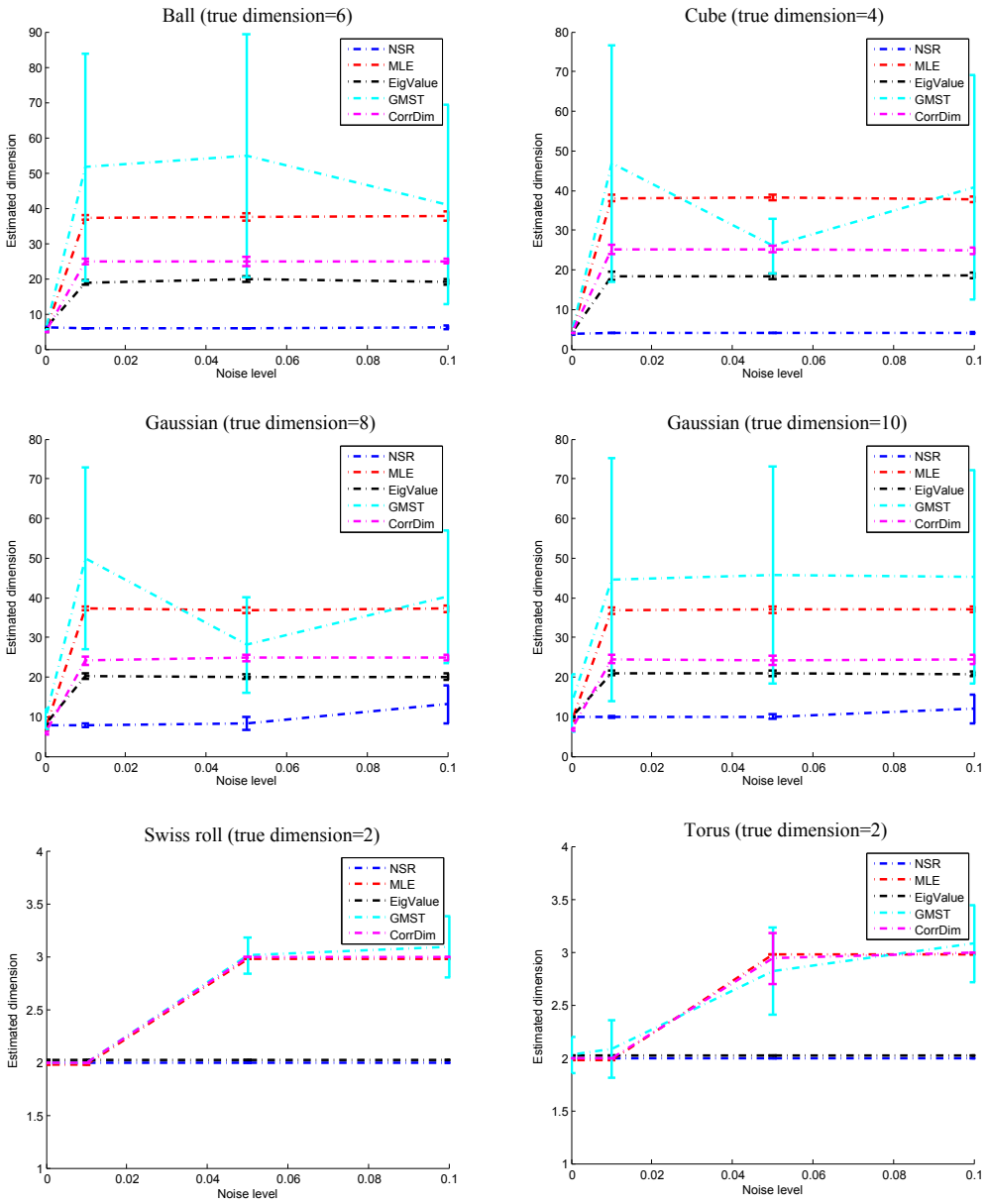
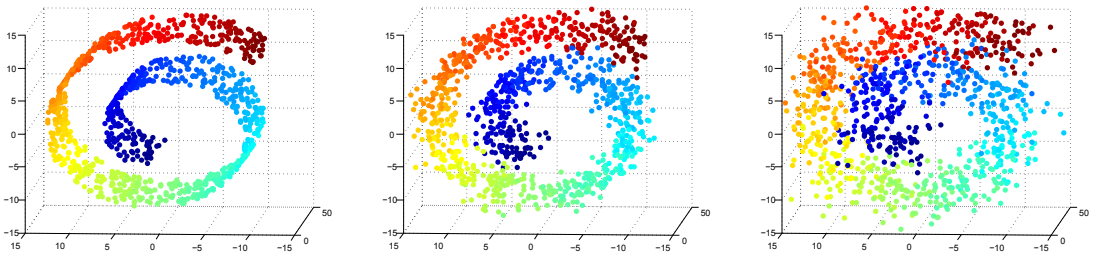


FIGURE 3.2: Latent-space dimensionality estimation on simulated data. We generated data from a 6–dimensional ball, a 4–dimensional cube, Gaussians of dimension 8 and 10, a 2-dimensional Swiss roll and a 2-dimensional torus. In the ball, cube and Gaussian cases, we embedded on 100 dimensions. In the Swiss roll and torus cases we embedded in three dimensions. In all cases, we added Gaussian noise with varying standard deviation (noise level in the plots). We compare our method (nonlinear spectral regression, NSR) with Maximum Likelihood Estimation (MLE), Eigenvalue thresholding (EigValue), Geodesic Minimum Spanning Tree (GMST) and Correlation Dimension (see [LB04]). Results are averaged over 100 runs, with standard deviations depicted.

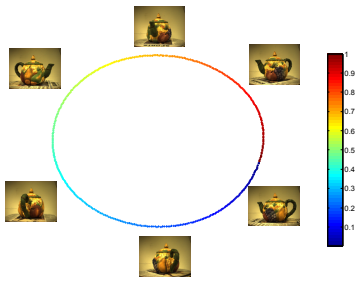


$\sigma_{\text{noise}} = 0$ (SNR=∞) $\sigma_{\text{noise}} = 0.05$ (SNR=20 dB) $\sigma_{\text{noise}} = 0.2$ (SNR=8 dB)

FIGURE 3.3: Data from a "Swiss roll" manifold, with varying amounts of added Gaussian noise. From left to right, the standard deviation σ_{noise} of the noise is 0 , 0.05 and 0.2. The color gradient indicates position, lengthwise, along the manifold. Note how, for the highest noise setting, the manifold structure is lost, due to short-circuits between different folds of the manifold. Any estimation algorithm is unlikely to recover the true intrinsic dimension.

new samples, because it learns two-way mappings between low and high-dimensional space. As shown in Figure 3.4, NSR is comparable to Nyström, for 25% and 50% out-of-sample data (5 runs, with different random data partitions); the metric we have adopted is the MSE between the embedding with the full training set and the embedding learned used only the in-sample data. An important distinction between the proposed NSR and Nyström is that the latter requires access to the training data when embedding new samples, while NSR does not (see Section 3.2.5; a full distribution is available for embedding new data, and here we present the mean). Also shown are the MSE results for embedding the MNIST (400 images per digit from 0-9) and the face datasets with NSR. We do not compare these with the Nyström method due to their more irregular and high-dimensional nature, which makes the Nyström approximation more unstable. To further examine the embedding associated with the MNIST data, example results are shown in Figure 3.5, with the embedding shown in two dimensions for digits 0-4.

We now present results on the face images considered in [RS00]. The images are grayscale with 20×28 pixels. Figure 3.6 shows a 2D view of the embedding computed by NSR. The model inferred 21 clusters, and an approximation to the



	MSE	
	50% out-of-sample	25% out-of-sample
Teapot (Nyström)	0.0292 ± 0.0152	0.0195 ± 0.0047
Teapot	0.0376 ± 0.0118	0.0165 ± 0.0069
MNIST	0.1075 ± 0.0285	0.0844 ± 0.0252
Frey faces	0.0632 ± 0.0184	0.0429 ± 0.0171
ISOMAP faces	0.0827 ± 0.0168	0.0582 ± 0.0125

FIGURE 3.4: Left: NSR embedding of teapot dataset based on all of the data; the color bar represents the true relative angle. Right: Performance of NSR on inferring latent features of held-out data, on four data sets; for the teapot data, we compare to Nyström's method [DM05].

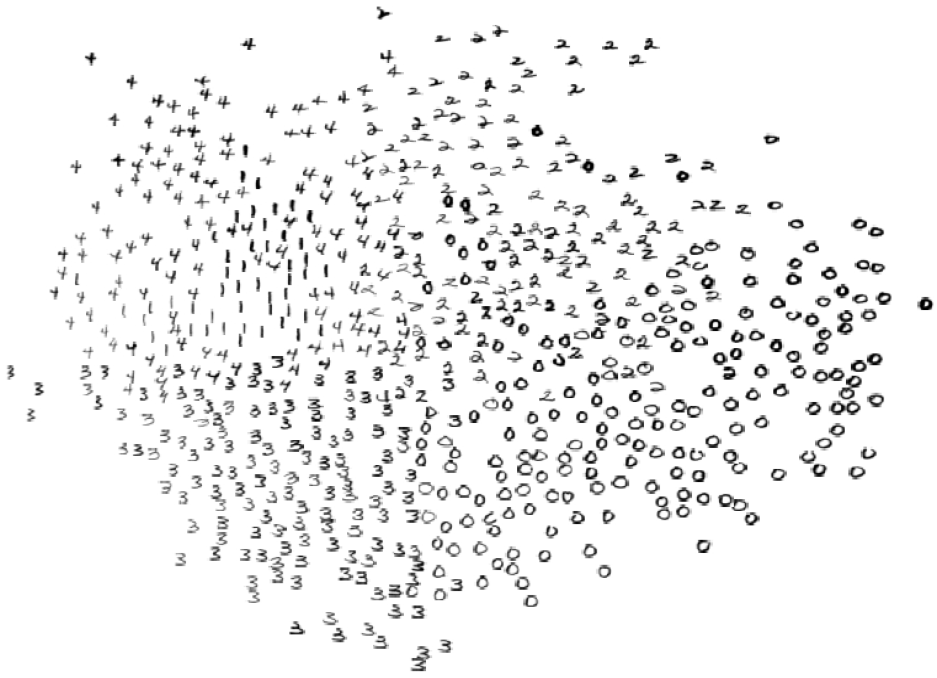


FIGURE 3.5: Embedding for digits "0"–"4" in two dimensions. Note how, for instance, digits "1" and "4" are embedded near each other.

posterior distribution of the latent-space dimensionality is also depicted in Figure 3.6. Further, in Figure 3.6 are shown NSR-generated *synthesized* face images along five selected cuts (A–E) in latent space, depicting smooth evolution of facial expressions. The synthesis of these faces is performed as discussed in Section 3.2.4; we present the expected synthesized high-dimensional image, based upon the cor-

responding position in the low-dimensional embedding space. The cuts are straight lines in 12 dimensions (we select the mean dimension from the aforementioned posterior on the dimensionality); the figure depicts 2D projections of the cuts from the 12-dimensional space. The endpoints of each cut are actual images from the training set, and the intermediate images are synthesized.

Considering cut A in Figure 3.6, which shows the subject with his tongue slowly sticking out, we show comparative results for alternative synthesis methods (top-right in Figure 3.6). The three alternative methods are: (*i*) mapping a latent feature vector to its nearest neighbor (NN) from the training set, and using the associated high-dimensional representation as the synthesis (consequently, all "synthesized" data actually come from the training set); (*ii*) performing SVD on the training data, and using the SVD coordinates as latent features, and the principal components to perform synthesis (denoted SVD); and (*iii*) linear interpolation (LI) in the high-dimensional space, using the endpoints of the line. Note that LI and NN require access to the high-dimensional data for synthesizing any new image, and LI never actually operates in the low-dimensional embedding space (of interest in the next section, when we consider synthesis of *dynamic* data). As expected, NN yields a non-continuous discretized synthesis, and SVD loses details in the face (the tongue is blurred). The LI results are comparable to those of the proposed method, NSR, but LI does not achieve our goal of a low-dimensional embedding. By contrast NSR yields effective synthesis from a low-dimensional embedding space, and the high-dimensional training data are not needed after the model is learned.

3.4.4 Dynamic analysis & synthesis

We consider motion-capture data available from `http://people.csail.mit.edu/ehsu/work/sig05stf` (termed MIT data) and from `http://mocap.cs.cmu.edu` (termed CMU data), as in [THR07, WFH08]. We obtained the animations by modifying code

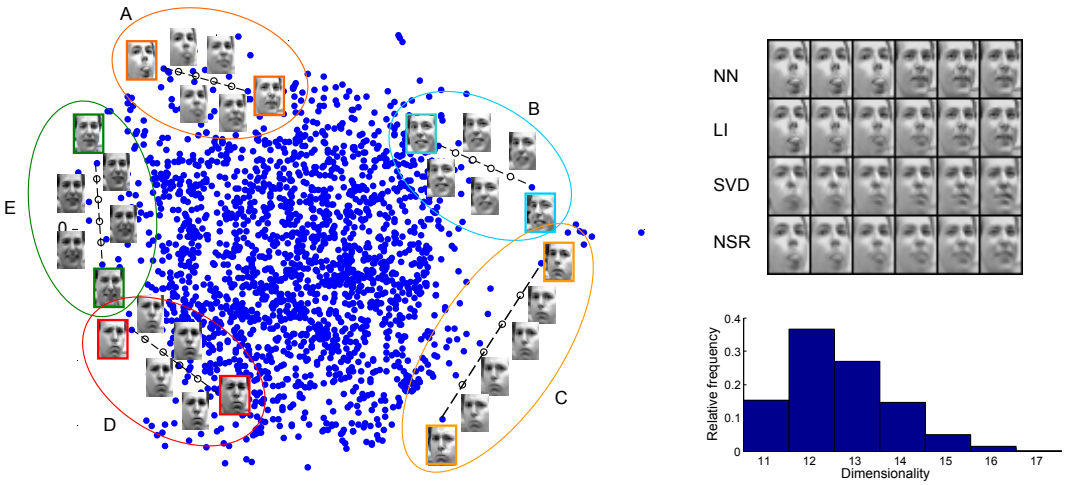


FIGURE 3.6: Face embedding and synthesis. **Left:** five straight line cuts (A–E) are shown in latent space, projected down to 2D. The endpoints of each cut (with colored borders) are images from the training set, while the intermediate images are synthesized. **Top right:** comparison of NSR with alternative methods for cut A, where the subject is slowly sticking his tongue out. From top to bottom, we show results using the nearest neighbors in the training set to the sample points on the cut (NN), the linear interpolation in high dimensions between endpoints (LI), SVD–based synthesis using 12 coordinates and our NSR–based synthesis, also using 12 coordinates. **Bottom right:** histogram of inferred dimensionality values from the posterior.

from `http://www.dcs.shef.ac.uk/~neil/mocap`. We have used 6 exercise routines from Subject 13 and 14 in the CMU data (as in [FSJW10]) and 4 sequences of walking and jogging from the MIT data. Each CMU frame is 62-dimensional, while the MIT data is 108-dimensional. The CMU and MIT data sources are analyzed separately; however, in each case all forms of motion are analyzed jointly, as discussed in Section 3.3.2.

When learning the embedding model, the latent dimension was truncated at $K = 30$ and the number of mixture components was truncated at $J = 60$; 2000 burn-in Gibbs samples were run, with 500 collection samples. For the dynamic model discussed in Section 3.3, the truncation level on the number of mixture components was set at $J^{dyn} = 80$. For the dynamic model we have considered 2000 Gibbs

burn-in iterations and 500 collection iterations. As an example, when analyzing the CMU data, the NSR embedding model infers 50 mixture components and a latent dimensionality of 8, and the associated dynamic model infers 58 mixture components; for the dynamic analysis the NSR embedding was implemented via diffusion [CL06], with similar results obtained via Isomap. The diffusion was performed using the radial-basis function kernel discussed in Section 3.2.2.

Each form of motion is characterized by a mixture model (Section 3.3.2), and the dynamic motion associated with each mixture component is defined by a local linear-regression model. We analyze all forms of dynamic data together, and using the method in Section 3.3.2 we cluster the different types of dynamic behavior (ideally, each inferred mixture component will represent a different form of dynamic motion). In Figure 3.7 we depict example dynamic data associated with five of the inferred mixture components; the model effectively learns and clusters different basic forms of motion. These results can be observed in greater detail by viewing the actual video sequences, which have been posted to `https://sites.google.com/site/npbnsr`. Similar motion-clustering results were presented in [FSJW10], but the model in [FSJW10] is not capable of synthesis.

We now consider dynamic-data *synthesis*. To provide a quantitative analysis, and to compare with other methods, we removed a contiguous set of 30-frames from the videos of two different subjects before dynamic training, and then computed the root mean-squared error (RMSE) on the held out frames, using the dynamic model to synthesize missing frames; we also computed the maximum RMSE difference between consecutive synthesized frames, to quantify the smoothness of the synthesized data. Results are averaged for 12 different windows (*i.e.*, frames 35-65, 36-66, etc., removed). The two subjects considered in this example corresponded to running motion, but results were similar for all other motions considered. As comparisons, we considered learning an embedding based on a factor-analysis model, rather than the

employed NSR mixture model. The purpose of this test is to examine the value of the nonlinear NSR embedding procedure, compared to linear FA (which is essentially statistical PCA). After performing the FA-based embedding, the dynamic motion was modeled exactly as used based on the NSR embedding, to provide a fair comparison. Additionally, we considered direct cubic spline interpolation in the high-dimensional space, thereby not explicitly learning a dynamic model. As observed in Table 3.2, the proposed method yields excellent performance relative to these alternatives. This is manifested not only in smaller RMSE, but also in smoother and more natural-looking motion (quantified via inter-frame differences); FA has reasonable RMSE but gross motion discontinuities (manifested by larger inter-frame differences), while the spline motion is smooth but very highly distorted relative to truth (high RMSE). We also tried, as a comparison, to do synthesis based on a dynamic model learned directly in the original high-dimensional space, without the intervening step of embedding to a low-dimensional latent space; this failed completely, based upon the limited training data available.

Table 3.2: RMSE for held-out 30-frame windows, and maximum squared norm of the difference between consecutive frames (higher values mean less smoothness). NSR is compared with factor analysis (FA) and spline interpolation.

	RMSE			Max inter-frame difference		
	NSR	FA	Spline	NSR	FA	Spline
Subject 1	31.11	37.68	81.38	53.54	78.01	46.89
Subject 2	27.57	35.99	75.19	57.08	80.02	48.25

We now employ the inferred dynamic model to generate motion automatically in the latent space, and then to project this back to the high-dimensional space for visualization (*i.e.*, computer-generate dynamic-motion synthesis). In Figure 3.8 we provide a small example of such synthesized motion, demonstrating the power of modeling the dynamics of multiple types of motion simultaneously (as discussed in Section 3.3.2). Multiple types of motion have the opportunity to share local

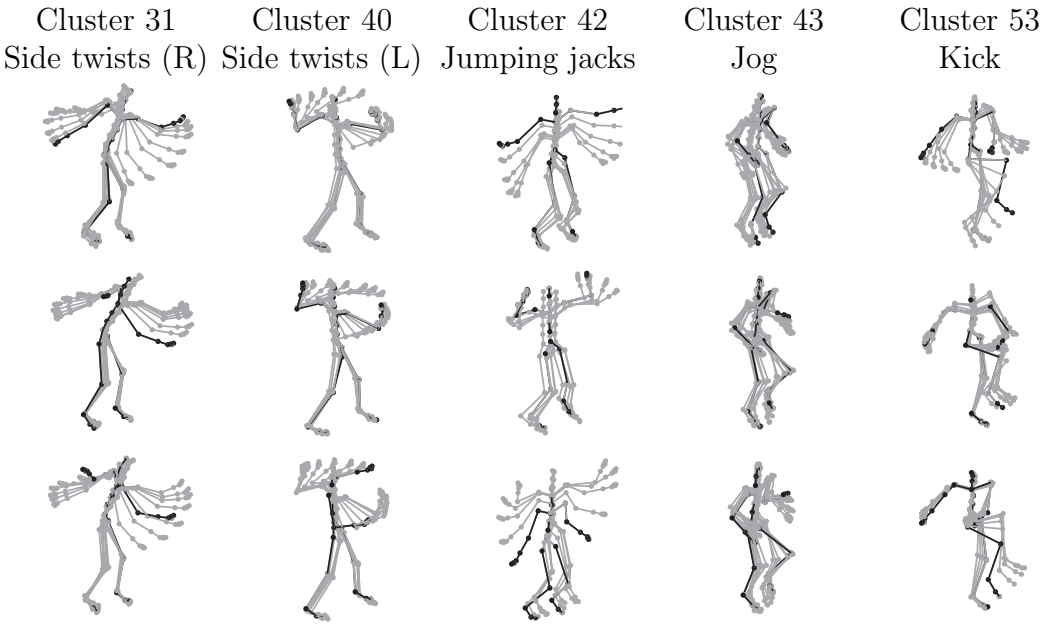


FIGURE 3.7: Clustering multiple dynamical behaviors, with dynamic modeling performed jointly. Each plot shows one 16-frame sequence from the training set, with the last frame in a darker color. Each column corresponds to the same most probable cluster. We illustrate five representative clusters, with the cluster index arbitrary. Analysis performed with the CMU data.

linear dynamic models (by sharing associated model parameters within the HDP). To generate the data in Figure 3.8, we initially synthesized dynamic data by using the learned walking model, and when this model moved into a part of latent space at which it shared dynamics with the running model, we turned over the dynamics to the running model. This allows us to synthesize a walking sequence followed by running; similar types of transitions may be manifested using the other forms of motion, assuming they share local dynamics. Figure 3.8 shows the synthesized smoothed transition from walking to running.

As a final example, in Figure 3.9 we show synthesized data for a limping sequence. This shows the range of dynamic motion that may be synthesized in the high-dimensional space, based upon dynamic models learned in the low-dimensional embedding space. The full video for these and other examples is at `https://sites.google.com/site/npbnsr`.

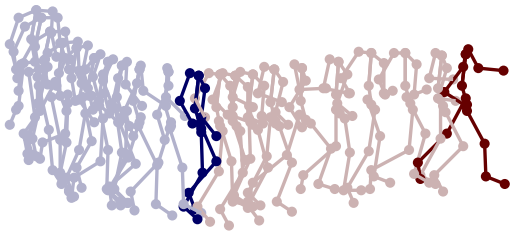


FIGURE 3.8: Fully synthesized motion sequence, with transition from walking (blue) to running (red). Model learning was performed using the MIT data.

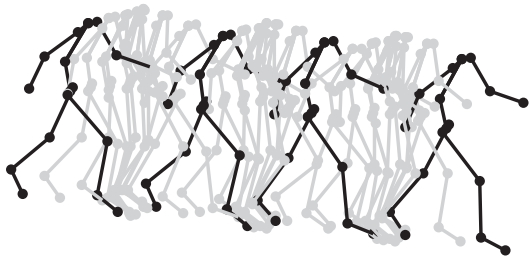


FIGURE 3.9: Fully synthesized motion sequence, showing limping behavior. Model learning was performed using the MIT data.

3.4.5 Brief discussion of computations

Space precludes providing explicit details of how the computations were performed. However, the complete hierarchical form of the models are posted at the aforementioned website. We also summarize there hyper-parameter settings, which were not optimized and are "routine" for models of this type. Because of the conjugate-exponential form of all aspects of the hierarchical models, all analysis is performed using analytic Gibbs update equations. Consequently, while the hierarchical models are relatively sophisticated, the detailed collapsed Gibbs inference is relatively routine. All computations were performed in (non-optimized) Matlab and were run efficiently on a PC (2.5 GHz clock). As examples of the computational cost, the teapot embeddings using all data required 4.4 hours, while the joint dynamic-model learning for 4 sequences of walking and jogging (MIT data) required 5.7 hours for

2500 iterations.

3.5 Conclusions

A new *statistical* spectral embedding framework is proposed, generalizing such spectral methods as diffusion and Isomap. This framework provides the ability to discard the training data when embedding new data, while also allowing synthesis of high-dimensional data from the embedding space. This procedure has been used to analyze high-dimensional dynamic data, with the nonlinear dynamics learned in the low-dimensional latent space. The proposed hierarchical dynamic model performs joint learning of dynamics from multiple types of motion, allowing the learning of shared structure. The model has also been demonstrated as a tool for analysis and synthesis of high-dimensional dynamic data. Finally, the method has proven an effective tool for estimating the dimensionality of a data set, even in the presence of substantial additive noise. The model has been implemented using Gibbs sampling, in which all update equations are analytic, and the models have been found to mix efficiently.

4

Topic Modeling with Nonparametric Markov Tree

4.1 Introduction

Latent Dirichlet Allocation (LDA) [BNJ03] is widely used to infer low-dimensional latent semantic information (topics) associated with a corpus of documents, where a topic is a distribution over words. While LDA constitutes a powerful modeling paradigm, and has served as the motivation for many subsequent models, a limitation of LDA and many such models is that within the prior no statistical dependencies are assumed between topics (each topic is drawn i.i.d. from a Dirichlet distribution). However, such statistical dependencies typically exist, for example a topic related to soccer and another topic related to football share many words, but these topics are not exactly the same; it is desirable to impose within the model this prior expectation of statistical dependencies between topics. To address this challenge, there has been recent interest in *hierarchical* topic models [AGJ10, BGJT03, BGJ10, GST07, Hof99, JMOB10]. In such settings one may view the model as inferring "meta-topics" that are constituted by integrating modular components ("sub-topics") within a hierarchy. For example, assume that ϕ_s is a probability vector representing a sub-topic, and

$\{\phi_s\}$ represents a finite set of such probability vectors; any convex combination of the $\{\phi_s\}$ may be viewed as constituting a meta-topic. Hierarchical models are typically based on modular elements like $\{\phi_s\}$, the inter-relationships between which are often constituted in a tree-based manner. In the context of the soccer/football illustration above, each may be a meta-topic, constituted by a convex combination of sub-topics from $\{\phi_s\}$; these two meta-topics will likely share some components $\{\phi_s\}$, but not all.

An important advantage of using such hierarchical and modular models is that different meta-topics (which may be defined, for example, by a branch of a tree) share components of the set $\{\phi_s\}$, and therefore there is a significant opportunity to borrow statistical strength efficiently across a corpus. While two meta topics may be distinct, they may share components of $\{\phi_s\}$, and therefore the available data are shared to a desirable extent when learning $\{\phi_s\}$.

In the nested Chinese restaurant process (nCRP) topic model [BGJ10], each node in the tree is characterized by a sub-topic, and a document is generated from one path (branch) through the tree, from the root to a leaf. In the tree-structured stick breaking process (TSSB) [AGJ10], each node has a unique distribution over topic and a document is generated from one node of the tree. These hierarchical topic models yield good performance. However, they sometimes may be too rigid. For example, in the nCRP model [BGJ10] there is a single root node, and children nodes may only have a single parent. This means that all descendent sub-topics from parent $p1$ must be distinct from the descendants of parent $p2$, if $p1 \neq p2$. Some of these distinct sets of children from different parents may be redundant, and this redundancy can be removed if a child can have more than one parent; this is one motivation of our proposed model. To get a sense of our model, and to observe its distinction from tree-based models such as that in [BGJ10], consider Figure 4.1. This figure depicts a small subset of the sub-topics inferred at two adjacent scales, for a real document

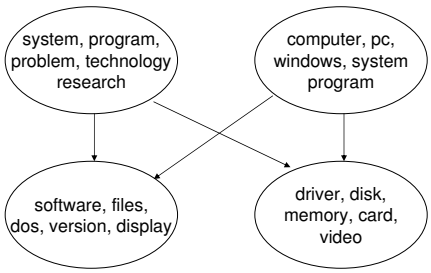


FIGURE 4.1: Illustrative example topics, inferred form the *20 Newsgroup* document corpus.

corpus, considered in detail when presenting results. A Markov transition process is inferred to move from sub-topics at one scale to those at the next scale, and from Figure 4.1 it is possible for a sub-topic to have two or more parents, linked to the parents in a statistical (Markov) sense. In Figure 4.1, the sub-topics at the bottom distinguish between "software" (left) and "hardware" (right); at the top layer the left sub-topic corresponds to general systems/technology, while the right sub-topic corresponds specifically to computers and PCs. By sharing children across multiple parent nodes, there is not a need to have children specialized to a distinct parent, and therefore potentially have nearly duplicate but decoupled children. This model flexibility is anticipated to enhance the sharing of statistical strength, in the sense discussed above. To our knowledge, none of the hierarchical and tree-based models developed previously have this flexibility.

Each of the sub-topics reflected in $\{\phi_s\}$ are probability vectors over the vocabulary. In most existing tree-based hierarchical models there is typically nothing within the model prior [AGJ10, BGJ10, GST07, Hof99] that places restrictions on these multiple probability vectors; hence, when drawing the ϕ_s from the prior, there may be significant duplication in word usage, in the sense that a particular word may be probable in many of the ϕ_s. This problem may be partially mitigated in the posterior for $\{\phi_s\}$, after analyzing the corpus; however, it is desirable to impose as

much structure as possible within the prior, such that there is less reliance on the data to infer anticipated phenomena. So motivated, within the proposed model we constitute a new framework, imposing that if a particular word is present in one or more sub-topics at a particular scale, then this word may not be used for sub-topics at scales deeper in the tree. Among other things, this removal of redundant usage of the same word at multiple scales aids in interpreting the multi-scale sub-topics inferred by the model.

We employ a retrospective sampler [PR08], within a stick-breaking representation of the Dirichlet process [Fer73] and related stick-breaking constructs [Set94]. In this setting the depth and width of the tree is unbounded, and hence the tree structure is inferred nonparametrically from the data. To the authors' knowledge, the form of the retrospective sampler employed in the proposed model is also new to topic modeling.

4.2 Proposed Model

4.2.1 Multi-scale Markov tree

We wish to model a corpus composed of D documents; the size of the vocabulary is V. We develop a hierarchical Bayesian model that infers a multi-scale topic construction. The lowest-level scale/resolution is $s = 1$, and we wish more detailed/specific words to be emphasized as one progresses deeper in the tree (*i.e.*, increasing s). Moreover, we wish to impose that if a word is utilized to constitute sub-topics at scale s', then this word is not reused in sub-topics at scales $s > s'$ (*i.e.*, deeper in the tree). The proposed model does not in general have a single root node, as in [BGJ10], and all children at scale $s + 1$ are connected statistically to all parents at scale s, via a Markov process. The depth of the tree, and the width (number of nodes) at each scale are inferred nonparametrically from the data.

For node t at scale s, there is an associated V-dimensional probability vector

over words, denoted $\boldsymbol{\phi}_{st}$, this representing a sub-topic. Further, when drawing word i from document d, w_{di}, there is a latent integer $c_{di} \geq 1$ defining which scale w_{di} is drawn from. The generative processes for $\{\boldsymbol{\phi}_{st}\}$ and $\{c_{di}\}$ are discussed below; here we describe the process by which we define the specific node at scale c_{di} from which word w_{di} is drawn.

The probability of utilizing each of the nodes (sub-topics) at the first scale $s = 1$ is defined by the document-dependent probability vector $\boldsymbol{\theta}_d$, drawn as

$$\boldsymbol{\theta}_d \sim \text{DP}(\eta, \boldsymbol{\alpha}) \ , \quad \boldsymbol{\alpha} \sim \text{Stick}(\lambda) \tag{4.1}$$

where $\text{DP}(\eta, \boldsymbol{\alpha})$ represents a Dirichlet process (DP) [Fer73] with base measure $\boldsymbol{\alpha}$ and real innovation parameter $\eta > 0$; the expression $\text{Stick}(\lambda)$ represents a stick-breaking process [Set94] with parameter λ, and the ith component of draw $\boldsymbol{\alpha}$ is $\alpha_i = V_i \prod_{h=1}^{i-1}(1 - V_h)$ with $V_h \sim \text{Beta}(1, \lambda)$. The aforementioned DP draw may be constituted as $\boldsymbol{\theta}_d \sim \sum_{j=1}^{\infty} \pi_j \delta_{\phi_j^*}$, with $\phi_j^* \sim \sum_{k=1}^{\infty} \alpha_k \delta_k$ and with $\boldsymbol{\pi} = (\pi_1, \pi_2, \dots)$ drawn $\boldsymbol{\pi} \sim \text{Stick}(\eta)$. The $\boldsymbol{\theta}_d$ are therefore drawn in a form related to the hierarchical DP (HDP) [TJBB06], with stick-breaking construction.

The node from which word w_{di} is drawn is manifested via a Markov process. Specifically, the generative process sequentially selects one node at each scale, up to scale c_{di}. When at node m at scale $s \geq 1$, the probability vector $\boldsymbol{p}_m^{(s)}$ defines the probability of which node is transitioned to at scale $s + 1$. This probability vector is drawn

$$\boldsymbol{p}_m^{(s)} \sim \text{Stick}(\zeta_s) \tag{4.2}$$

Drawing $\{\boldsymbol{p}_m^{(s)}\}$ in this manner for all nodes at scale s, a matrix $\mathbf{P}^{(s)}$ is defined, the mth column of which is $\boldsymbol{p}_m^{(s)}$. Note that the matrices $\{\mathbf{P}^{(s)}\}$ for scales $s \geq 1$ are assumed independent of the document d. If the model is truncated to only one scale, this model is essentially the previously developed HDP-based topic model [TJBB06].

The use of additional scales $s > 1$ are meant to capture finer details in the topics, adding flexibility by allowing incorporation of detail to the topics.

We will infer the number of required scales s to represent the corpus, using a retrospective sampler [PR08], as discussed in Section 4.2.4. Similarly, a retrospective sampler is also used to draw the $\boldsymbol{p}_m^{(s)} \sim \text{Stick}(\zeta_s)$, and therefore the number of nodes or sub-topics at each scale is also inferred. Hence we infer the depth of the tree, the width of each scale, and a Markovian statistical relationship between all parents at scale s and all children at scale $s+1$. Gamma hyperpriors are placed on λ and each ζ_s, and hence posterior distributions are also inferred for these parameters.

4.2.2 Node & scale-dependent word probabilities

Assume that through the aforementioned Markov process to scale c_{di}, node t is arrived at, and it is from this node that word w_{di} is drawn. Hence, word w_{di} is drawn from a multinomial distribution with parameter $\boldsymbol{\phi}_{c_{di}t}$. We now define a generative process for probability vectors $\{\boldsymbol{\phi}_{st}\}$, imposing that if a word has non-zero probability of occurring at scale s', then it has zero probability of occurring at scale $s > s'$.

At each scale $s \geq 1$ we define a V-dimensional binary vector $\boldsymbol{b}_s = (b_{s1}, \ldots, b_{sV})^T$. Each of the scale and node dependent probability vectors over words are drawn

$$\boldsymbol{\phi}_{st} \sim \text{Dirichlet}(\gamma \boldsymbol{b}_s) \tag{4.3}$$

If $b_{sv} = 0$, then word $v \in \{1, \ldots, |V|\}$ will have zero probability of being manifested at scale s (for all nodes t); *i.e.*, the vth component of $\{\boldsymbol{\phi}_{st}\}$ will be zero for all t. Therefore, within the model we impose that if the vth component of $\boldsymbol{b}_s$ is non-zero for a particular scale s, then the vth component of $\boldsymbol{b}_{s'}$ is zero for all $s' > s$. Specifically, the generative process is

$$p(b_{sv} = 1 | \boldsymbol{b}_{[s-1]v}) = 1(\boldsymbol{b}_{[s-1]v} = \boldsymbol{0})\tau_s, \tau_s \sim \text{Beta}(1, \psi_s) \tag{4.4}$$

where $\boldsymbol{b}_{[s-1]v} = (b_{1v}, \ldots, b_{s-1,v})^T$, $1(\cdot)$ is the indicator function, by convention $b_{0v} = 0$ for all $v \in \{1, \ldots, V\}$, τ_s is the conditional probability of adding a word to scale s given that it has not been added to any of the previous scales, and $\psi_s \geq 0$ is a hyperparameter controlling the distribution of words at scale s. One may wish to impose a separate prior for each scale-dependent parameter ψ_s, for example favoring smaller τ_s with increasing s (such that the $\{\boldsymbol{\phi}_{st}\}$ are sparser with increasing s, favoring more-detailed words). However, in our experiments we found that simply setting $\psi_s = \psi$ for all s worked well, with ψ drawn from a gamma distribution.

A distribution similar to the above Dirichlet($\gamma \boldsymbol{b}_s$) was used for language modeling in sparseTM [WB09a] and FTM [WWHB10]. However, the hierarchical construction specified above, for which words are not reused, is unique to the proposed model.

4.2.3 Generative process

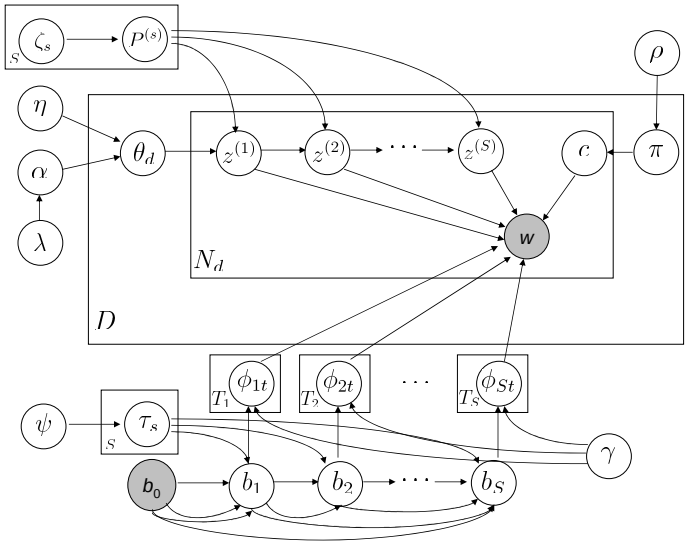


FIGURE 4.2: Graphical model representation for multi-scale Markov topic model.

Figure 4.2 provides a graphical depiction of the generative process. The generative

process of this model can be summarized as follow:

1. For each scale s, for each term v, draw term selector $b_{sv} \sim \text{Bernoulli}(1(\boldsymbol{b}_{[s-1]v} = \mathbf{0})\tau_s), \tau_s \sim \text{Beta}(1, \psi)$

2. For each topic $t \in 1, \ldots, T_s$ in scale s,

   (a) Draw topic distributions $\boldsymbol{\phi}_{st} \sim \text{Dirichlet}(\gamma \boldsymbol{b}_s)$

   (b) Draw transition matrix $\mathbf{P}^{(s)}$, the t-th column $\boldsymbol{p}_t^{(s)} \sim \text{Stick}(\zeta_s)$

3. Draw stick lengths $\boldsymbol{\alpha} \sim \text{Stick}(\lambda)$, which are the global distribution over topics

4. For document d

   (a) Draw distribution over topics for the first scale $\boldsymbol{\theta}_d \sim \text{DP}(\eta, \boldsymbol{\alpha})$

   (b) For the i-th word:

      i. Draw scale indicator $c_{di} \sim \text{Mult}(\boldsymbol{\pi}_d)$, $\boldsymbol{\pi}_d \sim \text{Stick}(\rho)$

      ii. Draw topic indicator $z_{di}^{(1)} \sim \text{Mult}(\boldsymbol{\theta}_d)$

      if $c_{di} \geq 2$, then for $s = 2, \ldots, c_{di}$ $z_{di}^{(s)} | (z_{di}^{(s-1)} = m) \sim \text{Mult}(\boldsymbol{p}_m^{(s-1)})$

      iii. Draw word $w_{di} | (c_{di} = s, z_{di}^{(s)} = t) \sim \text{Mult}(\boldsymbol{\phi}_{st})$

A gamma hyperprior is placed on ρ, and $\boldsymbol{\pi}_d$ defines the probability of using each of the scales in the tree, with the probability of scale usage document-dependent. Note that in principle $\boldsymbol{\pi}_d$ is an infinite-dimensional probability vector, and hence the number of scales is unbounded. As discussed in the next subsection, a retrospective sampler [PR08] is employed, and therefore the model infers the number of scales that are needed for representation of the corpus, just as a similar approach is employed to infer the number of nodes (sub-topics) at each scale.

Note that in our proposed model stick-breaking representations are employed in depth, and also in width, at each scale. In this sense the model is related to

the tree-structured stick-breaking model in [AGJ10]. However, in [AGJ10] an entire document is inferred to reside at one node in the tree, and in this sense the model may be viewed as yielding hierarchical clustering for documents. In the proposed model each node of the model corresponds to a sub-topic, as in [BGJ10]. However, unlike [BGJ10], children nodes may have multiple parent nodes, in a Markovian statistical sense, yielding a more-flexible construction with more sharing of sub-topics. The model in [JMOB10] has a parent-child tree-based construction similar to that in [BGJ10], but the finite tree must be specified, or inferred via cross-validation. The imposition, within the prior, of structure on word usage between scales is also unique to the proposed model.

4.2.4 Retrospective inference

In this subsection we describe the retrospective sampling scheme to learn S, the depth of the tree, and T_s, the width of the tree at each scale s. Let S be the learned depth of the tree in a given iteration and assume we have already obtain samples of $\{\{\boldsymbol{b}_s\}, \boldsymbol{\rho}, \{\boldsymbol{\phi}_{s,t}\}\}$ for $1 \leq s \leq S$. We learn S by updating each of the c_{di}'s in a Metropolis-Hastings step. When updating c_{di}, the proposed s' is generated from the following distribution

$$q_{di}(s') \propto \begin{cases} \pi_{s'} p(w_{di}|\boldsymbol{\phi}_{s',z_{di}^{(s')}}), & \text{for } s' \leq S \\ \pi_{s'} \mathcal{M}_{di}(S), & \text{for } s' > S \end{cases} \tag{4.5}$$

where $\mathcal{M}_{di}(S) = \max_{1 \leq s \leq S} \{p(w_{di}|\boldsymbol{\phi}_{s,z_{di}^{(s)}})\}$. The acceptance probability for the proposed s' is $\kappa_{di}(s, s')$, defined as

$$\kappa_{di}(s, s') = \begin{cases} 1, & \text{if } s' \leq S \text{ and } S' = S \\ \min\{1, \frac{\tilde{c}_{di}(S)\mathcal{M}_{di}(S')}{\tilde{c}_{di}(S')p(w_{di}|\boldsymbol{\phi}_{s,z_{di}^{(s)}})}\}, & \text{if } s' \leq S \text{ and } S' < S \\ \min\{1, \frac{\tilde{c}_{di}(S)p(w_{di}|\boldsymbol{\phi}_{s',z_{di}^{(s')}})}{\tilde{c}_{di}(S')\mathcal{M}_{di}(S)}\}, & \text{if } s' > S \end{cases} \tag{4.6}$$

where the normalizing constant $\tilde{c}_{di}(S) = \sum_{s=1}^{S} \pi_s p(w_{di}|\boldsymbol{\phi}_{s,z_{di}}) + \mathcal{M}_{di}(S)(1 - \sum_{s=1}^{S} \pi_s)$ and $S' = \max\{\max_{d' \neq d, i' \neq i}\{c_{d',i'}\}, s'\}$. The whole retrospective sampling procedure for c_{di} is summarized in Algorithm 1 in Appendix C. As shown in the algorithm, S is learned automatically by calculating the number of unique values of c_{di}'s. A similar retrospective sampling procedure can be developed for learning T_{s+1} by updating $\{\mathbf{P}^{(s)}\}$ [PC09a]. The rest parameters of the model are updated by Gibbs sampling and the update equations are provided in Appendix C.

4.3 Empirical Study

In the following experiments, if without other specifications, all the hyperparameters for the gamma distributions are set to Gamma$(10^{-3}, 10^{-3})$ and $\gamma = 1$, with no tuning performed on these parameters. These are the only parameters that need be set in the model, with an approximate posterior distribution estimated for all other parameters, based on the sampler. Within the sampler, we employed 2500 burn-in iterations, and we collected 500 samples after burn-in, taking every fifth sample to approximate the posterior. From the Bayesian perspective, the number of iterations may still be too small to ensure convergence, but in practice we find that they are large enough for achieving reasonable results.

4.3.1 Quantitative assessment

We compare the performance of our model to LDA [BNJ03] and nCRP [BGJ10]. We denote the proposed model as HMT, for "hierarchical Markov tree", and we present our model in two forms. The results denoted HMT-SD employ the scale dependency on word usage within sub-topics, as discussed in Section 2.2 (hence HMT-SD is our complete model). To examine the importance of imposing this scale-dependency to the word usage, we also consider HMT-NSD, in which no scale-dependency is imposed on the use of words within sub-topics; in this case $\boldsymbol{b}_s$ is all ones, for all

scales s. The models are examined on the following data sets:

- JCAM: a collection of 536 abstracts from the *Journal of the ACM* from 1987 to 2004 and the vocabulary size is 1539.[1]

- Psy. Review: a collection of 1281 abstracts from *Psychological Review* from 1967 to 2003 and the vocabulary size is 1971.

- 20 Newsgroups: A collection of 3000 documents, randomly selected articles from the *20 Newsgroups* data set; the vocabulary size is restricted to 5957 by a Porter stemmer.[2]

- NIPS: A collection of 1740 *NIPS* articles published from 1988 to 1999, and the vocabulary size is restricted to 7546 by a Porter stemmer.[3]

- Reuters: A collection of 3000 documents, randomly selected documents from the *Reuters*-21578 data set, and the vocabulary size is restricted to 1671 by a Porter stemmer.[4]

For all data sets, terms that appear in fewer than six documents were removed.

We first use the test set likelihood to evaluate the predictive performance of HMT. The common method to evaluate predictive performance in topic model is to use cross validation. Here we use five-fold cross validation. To calculate the conditional probability of the test set given the training set, we use the same method and same parameter settings as in nCRP [BGJ10] and VB-nCRP [WB09b]. Specifically, we

[1] http://www.cs.princeton.edu/∼blei/downloads/

[2] http://people.csail.mit.edu/jrennie/20Newsgroups/

[3] http://www.cs.nyu.edu/∼roweis/data.html

[4] http://kdd.ics.uci.edu/databases/reuters21578/

set the depth $S = 3$, $\eta = 1$ and use J samples and compute

$$p(\boldsymbol{w}_1^{\text{test}}, \ldots, \boldsymbol{w}_{D_{\text{test}}}^{\text{test}} | \boldsymbol{w}_1^{\text{train}}, \ldots, \boldsymbol{w}_{D_{\text{train}}}^{\text{train}}, \text{Model}) = \prod_{d,i} \frac{1}{J} \sum_{j=1}^{J} \sum_{s=1}^{S} 1(c_{di}^{(j)} = s) \sum_{t} \hat{\theta}_{dst}^{(j)} \psi_{stw_{di}}^{(j)} \tag{4.7}$$

where $\hat{\boldsymbol{\theta}}_{ds}^{(j)} = \mathbf{P}^{(s-1)^{(j)}} \ldots \mathbf{P}^{(1)^{(j)}} \boldsymbol{\theta}_d^{(j)}$. Larger log likelihood is better, and the log likelihood is computed by averaging across the collection samples. The mean test set log likelihood values and the standard deviations are shown in Table 4.1. For the LDA [BNJ03] results, similar to [BGJ10], we first run HMT-SD to obtain a posterior distribution over the number of topics, and then run LDA multiple times using the learned range of the number of topics in HMT-SD. The nCRP results are directly from [WB09b]. In each case, the HMT-SD achieves larger log likelihood than the three alternative models, significantly better than LDA. The performance of HMT-SD is similar to that of nCRP in *JACM* and does better in *Psychological Review*. As stated in nCRP [BGJ10], for a larger corpora the nCRP may be rigid for constraining to use only a single path, but HMT-SD provides more flexibility in selecting topics. From these results we also note that the imposition of structure on the probability of using words in sub-topics, as a function of scale s yields significant improvements. Specifically, both HMT-NSD and HMT-SD employ the same tree structure, while the former does not impose structure within the prior on word usage in scale-dependent sub-topics.

Table 4.1: Test set per-word log likelihood for the five data sets, and four algorithms. The HMT-SD model is that proposed here, and HMT-NSD is a comparison simplification. The LDA model is from [BNJ03] and the nCRP model is from [BGJ10].

	JCAM	Psy. Review	20 Newsgroup	NIPS	Reuters
LDA	-13.6237±0.0159	-14.6483±0.0192	-15.1791±0.0149	-16.2404±0.0183	−13.7142±0.0168
nCRP	-5.3922±0.0052	-5.7834±0.0149	*	*	*
HMT-NSD	-5.6048±0.0059	-5.7530±0.0153	-6.6949±0.0175	-6.8962±0.0178	-5.6575±0.0150
HMT-SD	**-5.2770±0.068**	**-5.4734±0.141**	**-6.2952±0.0135**	**-6.4665±0.0194**	**-5.3746±0.0105**

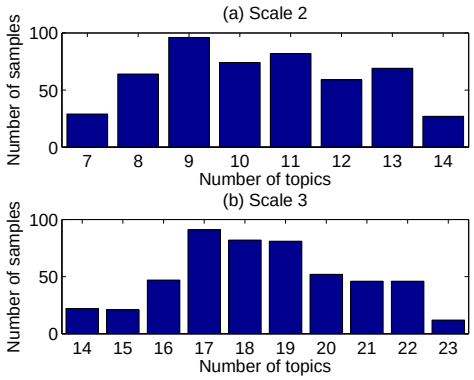


FIGURE 4.3: Approximate posterior distribution (histogram) on the number of topics at scales $s = 2$ and $s = 3$, for the proposed model considering the *20 Newsgroup* data.

4.3.2 Model structure and parsimony

Within the MCMC sampler, each collection sample yields a unique topic tree. To get a sense of the variety of models that are manifested via these collection samples, in Figure 4.3 we show a histogram of the number of topics at scales $s = 2$ and $s = 3$ for the *20 Newsgroup* data set; similar results were found for the other data sets. Note that these numbers of topics at each scale were inferred efficiently via the retrospective sampler, and a similar distribution is found with respect to the depth of the tree.

To further examine the properties of the model, we examine the number of words found in each topic in different scale, on average. The results in Figure 4.4, for the aforementioned models and data sets, indicate that the HMT-SD model yields a more parsimonious usage of words across topics. In Figure 4.4 we present mean results, as well as the standard deviation. In each case, the number of terms per topic in HMT-SD is significantly less than that of HMT-NSD. This is attributed to the HMT-SD prior that removes redundancy of words at multiple scales (although there can be redundancy *within* a single scale), and therefore the topics tend to be more focused and less redundant. In addition to aiding the ability to interpret inferred topics, this

construction improves quantitative log likelihood results, as discussed above.

4.3.3 Interpreting the learned tree

In Figure 4.5 we present an example topic tree inferred from the proposed model. In this analysis a single root node was employed, meant to capture the ubiquitous and non-informative words. The transition probabilities from the root node to the second layer are document-dependent, and all other transition probabilities between layers are document-independent. Within Figure 4.5, which corresponds to a typical collection sample, we present the top-five most-probable words within each sub-topic, and the arrows represent non-zero transition probabilities. Note that sub-topics at layer $s = 3$ often have multiple parents, in a statistical (Markovian) sense, this representing a unique component of the proposed model.

Careful examination of Figure 4.5 demonstrates that several interesting relationships are inferred between sub-topics at different scales. For example there appear at layer $s = 3$ to be sub-topics on hockey and baseball, and each of these share a parent at layer $s = 2$ that appears to capture sports in a generic sense. Another interesting example concerns a sub-topic at scale $s = 2$ that focuses on systems and technology, and this is connected to children sub-topics at layer $s = 3$ focusing (separately) on bikes, cars, space travel, electronic devices, software, and computer hardware. Similar trees are manifested by the model in [BGJ10] (the code for [BGJ10] was unavailable at the time of writing to do a comparison). As discussed above, two unique aspects of the proposed model are the opportunity for a sub-topic to have more than one parent, in a statistical sense, and also the lack of word duplication between scales. Figure 4.1 presents a zoom-in taken from Figure 4.5, in which the case of multiple parents is observed.

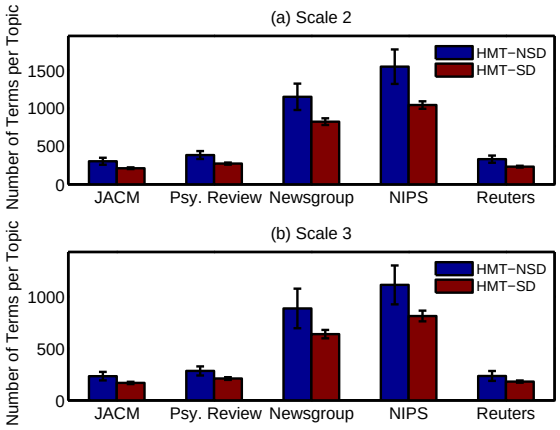


FIGURE 4.4: Number of terms per topic at scales $s = 2$ and $s = 3$ for HMT-SD and HMT-NSD.

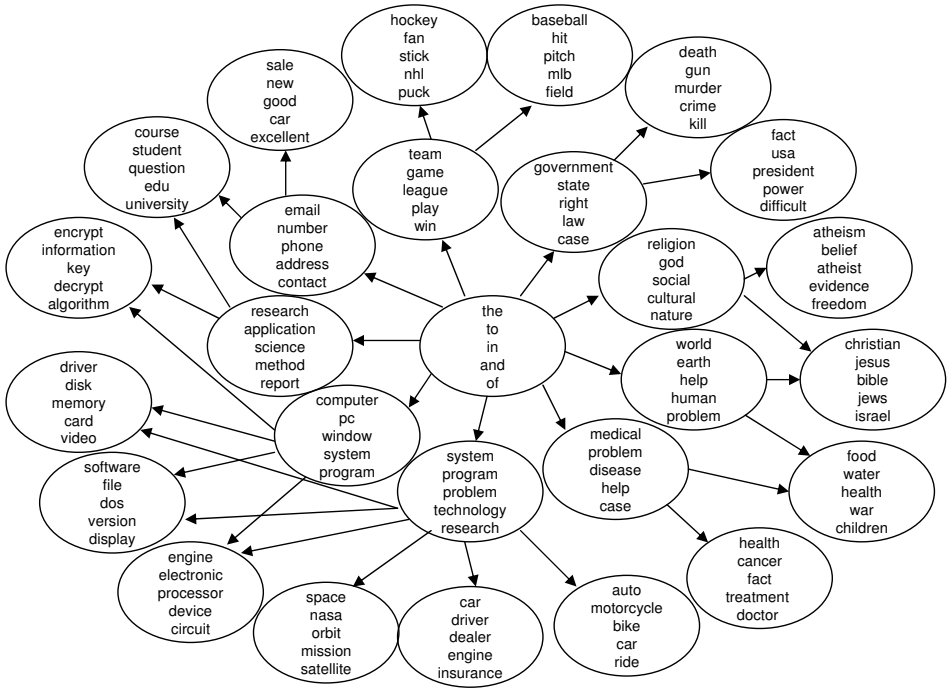


FIGURE 4.5: Example tree inferred from *20 Newsgroup* data set; these results correspond to one (typical) collection sample.

4.4 Conclusions

A new hierarchical tree-based topic model has been presented. The model removes redundancies in two ways: (i) sub-topics may have multiple parents, thereby yielding a flexible, statistical branching structure; and (ii) if a word is used in a sub-topic at a particular scale, it is not re-used at deeper scales. A retrospective sampler is employed to infer both the tree depth and width (the width is scale-dependent). State-of-the-art results are achieved on five data sets, with comparisons against recently developed related models.

5

Conclusion and Future Work

In the real world, high-dimensional data can often be represented by some sort of sparse or low-dimensional latent model. Learning such a latent model reduces the quantity of data required by the system and achieves better interpretation for the data. In this thesis, nonparametric Bayesian techniques are employed to learn the sparse or low-dimensional latent models and infer the appropriate dimensionality in the latent space simultaneously. Three hierarchical models are proposed for images, dynamic data, and documents, respectively. Encouraging results are achieved, with comparison to other state-of-the-art approaches.

The contribution of this thesis can be summarized as follows:

- Nonparametric Bayesian techniques are considered for learning dictionaries for sparse image representations and recovery of imagery based upon compressive measurements. A truncated beta-Bernoulli process is employed as a prior for learning the dictionary and image recovery, and therefore the number of dictionary elements and their relative importance may be inferred nonparametrically. In addition, the spatial inter-relationships between different components in images are exploited by use of the Dirichlet process and a pro-

bit stick-breaking process. Finally, the compressive-measurement projections are also optimized for the learned dictionary.

- Hierarchical Bayesian methods are employed to learn a reversible statistical embedding. The proposed approach allows one to discard the training data when embedding new data, allows synthesis of high-dimensional data from the embedding space, and provides accurate estimation of the latent-space dimensionality. A nonlinear statistical dynamic model is then learned in the embedding space, allowing analysis, modeling and synthesis of dynamic high-dimensional data. Further, a generalized hierarchical Dirichlet process is used to build a hierarchical dynamic model, allowing the sharing of statistical strength between multiple high-dimensional dynamic data sources, while also allowing inference of inter-relationships between such data.

- Nonparametric Bayesian techniques is used to developed a new hierarchical tree-based topic model. The model has two unique attributes: (i) a child node in the tree may have multiple parents, with the goal of eliminating redundant sub-topics deep in the tree; and (ii) A word-scale-dependent prior is proposed to remove redundant usage of words at multiple scales. The depth and width of the tree are unbounded within the prior, with a retrospective sampler employed to adaptively infer the appropriate tree size based upon the corpus under study.

These contributions motivate several directions for future research:

- For dictionary learning, additional prior information beyond sparseness may be imposed. It is of interest to explore the similarity, temporal information, and/or spatial distances of the data for dictionary learning. Successful examples include covariate-dependent dictionary learning for denoising and inpainting, structured-sparse dictionary learning for compressive sensing [HZM09] and

hierarchical tree-based dictionary learning for text and images [JMOB10], and so on.

- For embedding, further research may focus on experimenting with a Pitman–Yor construction in lieu of the Dirichlet process. For the dynamic examples, we intend to examine the value of learning the embedding and dynamic model jointly, which also may be done with analytic Gibbs sampling (although for the motion-capture data considered thus far, learning the dynamic model as a separate step appears to be effective, as found in [WFH08] using a different model). Another possible application for these models is to synthesize music.

- For hierarchical topic model, documents' side information may be utilized to build a supervised hierarchical topic model for discovering predictive low dimensional representations of documents. In addition, the statistical relationship between the parent and children of the HMT model is unchanged once learned. But the subtopics and their relationship may evolve over time. Therefore, time-evolving information may also be exploited to build a hierarchical topic model, with the subtopics and the Markovian transition matrices associated with time periods.

Appendix A

Summary of Update Equations for BPFA Model

A.1 Update Equations for Dictionary Learning

The MCMC updated equations for BPFA model are given below.

- Sample d_k, $k = 1, 2, \ldots, K$

$$p(d_k|-) \propto \mathcal{N}(\mu_{d_k}, \Sigma_{d_k}) \tag{A.1}$$

where

$$\begin{aligned} \Sigma_{d_k} &= (P + \gamma_\epsilon \sum_{i=1}^{N} z_{ik}^2 s_{ik}^2)^{-1} \\ \mu_{d_k} &= \gamma_\epsilon \Sigma_{d_k} \sum_{i=1}^{N} z_{ik} s_{ik} [x_i - \mathbf{D}^{-k}(z_i^{-k} \circ s_i^{-k})] \end{aligned}$$

- Sample γ_ϵ

$$p(\gamma_\epsilon|-) \propto Gamma(c + PN/2, d + \sum_{i=1}^{N} \|x_i - \mathbf{D}(z_i \circ s_i)\|_2^2/2) \tag{A.2}$$

- Sample γ_s

$$p(\gamma_s|-) \propto Gamma(e + KN/2, f + \sum_{i=1}^{N} \|s_i\|_2^2/2) \tag{A.3}$$

- Sample $s_{ik},\ \ i = 1, 2, \ldots, N, k = 1, 2, \ldots, K$

$$p(s_{ik}|-) \sim \mathcal{N}(\mu_{s_{ik}}, \sigma_{s_{ik}}) \tag{A.4}$$

where

$$\begin{aligned} \sigma_{s_{ik}} &= (\gamma_s + \gamma_\epsilon z_{ik}^2 d_k^T d_k)^{-1} \\ \mu_{s_{ik}} &= \gamma_\epsilon \sigma_{s_{ik}} z_{ik} d_k [x_i - \mathbf{D}^{-k}(z_i^{-k} \circ s_i^{-k})] \end{aligned}$$

- Sample $z_{ik},\ \ i = 1, 2, \ldots, N, k = 1, 2, \ldots, K$

$$p(z_{ik}|-) \propto Bernoulli(\frac{t_1}{t_1 + t_2}) \tag{A.5}$$

where

$$t_1 = \pi_{ik} \exp\{-\frac{\gamma_s}{2}[s_{ik}^2 d_k^T d_k - 2 s_{ik} d_k^T (x_i - \mathbf{D}^{-k}(z_i^{-k} \circ s_i^{-k}))]\}, t_2 = 1 - \pi_{ik}$$

- Sample $\pi_k, k = 1, 2, \ldots, K$

$$p(\pi_k|-) \propto Beta(\frac{a}{K} + \sum_{i=1}^{N} z_{ik}, \frac{b(K-1)}{K} + N - \sum_{i=1}^{N} z_{ik}) \tag{A.6}$$

A.2 Update Equations for DP-BPFA CS

The posterior distributions of the model parameters $\mathbf{\Theta}$ are inferred from observed data – CS measurements $\boldsymbol{y}$, via the VB inference [Bea03], which converges fast and is computationally efficient. The VB inference employs a set of distributions $q(\mathbf{\Theta})$ to approximate the true posterior distributions $p(\mathbf{\Theta}|\boldsymbol{y})$ by minimizing the Kullback-Leibler divergence, $KL(q(\mathbf{\Theta})\|p(\mathbf{\Theta}|\boldsymbol{y}))$, and uses a lower bound $\mathcal{F}$ to approximate

the true log-likelihood of the model $\log p(\boldsymbol{y}|\boldsymbol{\Theta})$. The algorithm iteratively updates $q(\boldsymbol{\Theta})$ so that $\mathcal{F}$ approaches to $\log p(\boldsymbol{y}|\boldsymbol{\Theta})$ until convergence. For DP-BPFA, $\boldsymbol{\Theta} = \{s, z, \gamma_s, \gamma_\epsilon, \pi, V, \alpha, r\}$. The update equations for the posterior distributions of $\boldsymbol{\Theta}$ are as follows:

- Update $q(s_i),\ \ i = 1, 2, \ldots, N$

$$q(s_i) = \mathcal{N}(s_i; \mu_i, \Sigma_i) \tag{A.7}$$

  where

$$\begin{aligned} \Sigma_i &= [\langle\gamma_s\rangle + \langle\gamma_\epsilon\rangle\langle diag(z_i)\Phi^T\Phi diag(z_i)\rangle]^{-1} \\ \mu_i &= \Sigma_i\langle\gamma_\epsilon\rangle diag(\langle z_i\rangle)\Phi^T y_i, \end{aligned}$$

- Update $q(\gamma_s),\ \ i = 1, 2, \ldots, N$

$$q(\gamma_s) = Gamma(c, d) \tag{A.8}$$

  where

$$c = c_0 + \frac{1}{2}NK, \quad d = d_0 + \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{K}\langle s_{ik}^2\rangle$$

- Update $q(\gamma_\epsilon)$

$$q(\gamma_\epsilon) = Gamma(g, h) \tag{A.9}$$

  where

$$g = g_0 + nN/2, \quad h = h_0 + \sum_{i=1}^{N}\langle\|y_i - \Phi(z_i \circ s_i)\|_2^2\rangle$$

- Update $q(\alpha)$

$$q(\alpha) = Gamma(e, f) \tag{A.10}$$

  where

$$e = e_0 + N_L - 1, \quad f = f_0 - \sum_{l=1}^{N_L-1}[\psi(\tau_{2l}) - \psi(\tau_{1l} + \tau_{2l})]$$

- Update $q(r_i = l) = m_{il},\ \ i = 1, 2, \ldots, N, l = 1, 2, \ldots, N_L$

$$m_{il} = \frac{\exp(\lambda_{il})}{\sum_{l=1}^{N_L} \exp(\lambda_{il})} \tag{A.11}$$

where

$$\begin{aligned} \lambda_{il} \quad = \quad & \sum_{t=1}^{l-1} [\psi(\tau_{2t}) - \psi(\tau_{1t} + \tau_{2t})] + \psi(\tau_{1l}) - \psi(\tau_{1l} + \tau_{2l}) + \\ & \sum_{k=1}^{K} [\langle z_{ik} \rangle \langle \log \pi_{lk} \rangle + (1 - \langle z_{ik} \rangle) \langle \log(1 - \pi_{lk}) \rangle] \end{aligned}$$

- Update $q(V_l),\ \ l = 1, 2, \ldots, N_L - 1$

$$q(V_l) = Gamma(\tau_{1l}, \tau_{2l}) \tag{A.12}$$

where

$$\tau_{1l} = 1 + \sum_{i=1}^{N} m_{il}, \quad \tau_{2l} = e/f - \sum_{i=1}^{N} \sum_{t=l+1}^{N_L} m_{it}$$

- Update $q(\pi_{lk}), k = 1, 2, \ldots, K, l = 1, 2, \ldots, N_L$

$$q(\alpha_l) = Gamma(\tau_{3lk}, \tau_{4lk}) \tag{A.13}$$

where

$$\tau_{3lk} = a/K + \sum_{i=1}^{N} m_{il} \langle z_{ik} \rangle, \quad \tau_{4lk} = b(K-1)/K + \sum_{i=1}^{N} m_{il} (1 - \langle z_{ik} \rangle)$$

- Update $q(z_{ik}),\ \ i = 1, 2, \ldots, N, k = 1, 2, \ldots, K$

$$q(z_{ik}) = Bernoulli(t_{ik}) \tag{A.14}$$

where

$$
\begin{aligned}
t_{ik} &= [1+\exp\{\sum_{l=1}^{N_L} m_{il}\psi(\tau_{4lk}) - \sum_{l=1}^{N_L} m_{il}\psi(\tau_{4lk}) \\
&\quad + \frac{\langle\gamma_\epsilon\rangle}{2}[\langle s_{ik}^2\rangle\Phi_k^T\Phi_k - 2\langle s_{ik}\rangle\Phi_k^T(y_i - \sum_{\substack{l=1\\ l\neq k}}^{N}\Phi_l\langle z_{il}\rangle\langle s_{il}\rangle)]\}]^{-1}
\end{aligned}
$$

The angle bracket $\langle\rangle$ represents the expectation with respect to the random variable in it. They can be evaluated as:

$$
\begin{array}{lll}
\langle z_{ik}\rangle = t_{ik}, & \langle s_{ik}\rangle = \mu_{ik}, & \langle\gamma_\epsilon\rangle = g/h, \\
\langle z_{ik}^2\rangle = t_{ik}, & \langle s_{ik}^2\rangle = \mu_{ik}^2 + \sigma_{ik}^2, & \langle\gamma_s\rangle = c/d,
\end{array}
$$

where ψ is the *digamma* function [Bea03] defined as $\psi(x) = \frac{\partial}{\partial x}\ln\Gamma(x)$.

Appendix B

Summary of Update Equations for NSR and Dynamic Model

B.1 Update Equations for NSR Model

The Gibbs sampling inference algorithm for the NSR model can be summarized as follows:

- Sample the cluster index $z(i)$ from

$$p(z(i) = j|-) \propto w_j \mathcal{N}(\boldsymbol{A}_j \boldsymbol{\Lambda}(\boldsymbol{x}_i - \hat{\boldsymbol{\mu}}_j) + \boldsymbol{\mu}_j, \alpha_{0j}^{-1} \boldsymbol{I}_D) \mathcal{N}(x_i; \hat{\boldsymbol{\mu}}_j, \beta_j^{-1} \boldsymbol{I}_K) \quad \text{(B.1)}$$

  After normalization across z, $p(z(i) = j|-)$ becomes a Multinomial distribution.

- Sample the DP concentration parameter α from

$$p(\alpha|-) = \text{Gamma}\left(\alpha; g_0 + J - 1, h_0 - \sum_{l=1}^{J-1} \log(1 - V_l)\right) \quad \text{(B.2)}$$

- Sample the DP stick weight V_j, $j = 1, 2, \cdots, J-1$ and $V_J = 1$ from

$$p(V_j|-) = \text{Beta}\left(V_j; 1 + \sum_{i:z(i)=j} 1, \alpha + \sum_{i:z(i)>j} 1\right) \tag{B.3}$$

- Sample $\boldsymbol{b}_k$ and $\boldsymbol{\lambda}_k$

$$\begin{aligned} p(b_k, \lambda_k|-) &\propto \text{Bernoulli}(b_k; \pi_k)\mathcal{N}(\lambda_k; 0, 1) \\ &\quad \prod_{i=1}^{M} \mathcal{N}\left(\boldsymbol{y}_i^{-k}; \boldsymbol{A}_{z(i)k}(x_{ik} - \hat{\mu}_{z(i)})b_k\lambda_k, \alpha_{0z(i)}^{-1}\boldsymbol{I}\right) \end{aligned}$$

with $\boldsymbol{y}_{ti}^{-k} \triangleq \boldsymbol{y}_i - \sum_{m\neq k} \tilde{\boldsymbol{A}}_{z(i)m}(x_{ik} - \hat{\mu}_{z(i)})b_k\lambda_k - \boldsymbol{\mu}_{z(i)}$. Thus we can sample $\boldsymbol{b}_k$ and $\boldsymbol{\lambda}_k$ from

$$\begin{aligned} p(b_k|-) &= \text{Bernoulli}(b_k; \tilde{\pi}_k) \\ p(\lambda_k|b_k, -) &= b_k\mathcal{N}(\lambda_k; \delta_k, \gamma_k) + (1 - b_k)\mathcal{N}(\lambda_k; 0, 1) \end{aligned} \tag{B.4}$$

where

$$\begin{aligned} \log\frac{\tilde{\pi}_k}{1-\tilde{\pi}_k} &= \log\frac{\pi_k}{1-\pi_k} + \frac{1}{2}\log\gamma_k + \frac{\delta_k^2}{2\gamma_k} \\ \gamma_k &= (1 + \sum_{i=1}^{M} \alpha_{0z(i)}\boldsymbol{A}_{z(i)k}^{\top}\boldsymbol{A}_{z(i)k}(x_{ik} - \hat{\mu}_{z(i)k})^2)^{-1} \\ \delta_k &= \gamma_k(\sum_{i=1}^{M} \alpha_{0z(i)}\boldsymbol{A}_{z(i)k}^{\top}\boldsymbol{y}_i^{-k}(x_{ik} - \hat{\mu}_{z(i)k})) \end{aligned}$$

- Sample the Bernoulli parameter $\boldsymbol{\pi}$ from

$$p(\pi_k|-) = \text{Beta}(\pi_k; a_0/K + b_k, b_0(K-1)/K + 1 - b_k) \tag{B.5}$$

- Sample the mean vector for each cluster $\boldsymbol{\mu}_j$ from

$$p(\boldsymbol{\mu}_j|-) = \mathcal{N}(\boldsymbol{\mu}_j; \boldsymbol{\varsigma}_j, \boldsymbol{\Gamma}_j) \tag{B.6}$$

where

$$\boldsymbol{\Gamma}_j = (\gamma + \alpha_{0j} \sum_{i:z(i)=j} 1)^{-1} \boldsymbol{I}_D; \quad \boldsymbol{\varsigma}_t = \boldsymbol{\Gamma}_j(\alpha_{0j} \sum_{i:z(i)=j} (\boldsymbol{y}_i - \boldsymbol{A}_j \text{diag}(\boldsymbol{b} \circ \boldsymbol{\lambda})(x_i - \hat{\mu}_j)))$$

- Sample the factor loading matrix $\boldsymbol{A}_j$ from

$$p(\boldsymbol{A}_{jk}|-) \propto= \mathcal{N}(\boldsymbol{A}_{jk}; \boldsymbol{\varrho}_{jk}, \boldsymbol{\Xi}_{jk}) \tag{B.7}$$

where $\boldsymbol{A}_{jk}$ denotes the kth row of matrix $\boldsymbol{A}_j$ and

$$\begin{aligned} \boldsymbol{\Xi}_{jk} &= (\zeta \boldsymbol{I}_K + \alpha_{0j} \text{diag}(\boldsymbol{b} \circ \boldsymbol{\lambda}) \sum_{i:z(i)=j} (x_i - \hat{\mu}_j)(x_i - \hat{\mu}_j)^\top \text{diag}(\boldsymbol{b} \circ \boldsymbol{\lambda}))^{-1} \\ \boldsymbol{\varrho}_{jt} &= \boldsymbol{\Xi}_{jk}(\alpha_{0j} \text{diag}(\boldsymbol{b} \circ \boldsymbol{\lambda}) \sum_{i:z(i)=j} (x_i - \hat{\mu}_j)(y_{ik} - \mu_{jk})) \end{aligned}$$

- Sample the precision of the column of factor loading ζ from

$$p(\zeta|-) = \text{Gamma}(\zeta; \tau_1, \tau_2) \tag{B.8}$$

with

$$\tau_1 = \tau_{10} + \frac{KJN}{2}; \quad \tau_2 = \tau_{20} + \frac{1}{2} \sum_{j=1}^{J} \sum_{k=1}^{K} \boldsymbol{A}_{jk}^T \boldsymbol{A}_{jk}$$

- Sample the precision of the additive noise α_{0j} from

$$p(\alpha_{0j}|-) = \text{Gamma}(\alpha_{0j}; c_j, d_j) \tag{B.9}$$

with

$$c_j = c_0 + \frac{D}{2} \sum_{i:z(i)=j} 1; \quad d_j = d_0 + \frac{1}{2} \sum_{k=1}^{D} \sum_{i:z(i)=j} \|y_{ik} - \boldsymbol{A}_{jk} \text{diag}(\boldsymbol{b} \circ \boldsymbol{\lambda})(x_i - \hat{\mu}_j) - \mu_{jk}\|_2^2$$

- Sample $\hat{\boldsymbol{\mu}}_j$, $m = 1, 2 \ldots, N$ from

$$p(\hat{\boldsymbol{\mu}}_j|-) = \mathcal{N}(\hat{\boldsymbol{\mu}}_j; \boldsymbol{\mu}_{\hat{\mu}_j}, \boldsymbol{\Sigma}_{\hat{\mu}_j}) \tag{B.10}$$

with

$$\begin{aligned}
\boldsymbol{\Sigma}_{\hat{\mu}_j} &= \left((\alpha_{0j} \text{diag}(\boldsymbol{b} \circ \boldsymbol{\lambda}) \boldsymbol{A}_j^\top \boldsymbol{A}_j \text{diag}(\boldsymbol{b} \circ \boldsymbol{\lambda}) + \boldsymbol{\Sigma}_j^{-1}) \sum_{i:z(i)=j} 1 + \xi \boldsymbol{I}_d \right)^{-1} \\
\boldsymbol{\mu}_{\hat{\mu}_j} &= \boldsymbol{\Sigma}_{\hat{\mu}_j} \left(\sum_{i:z(i)=j} \boldsymbol{\Sigma}_j^{-1} \boldsymbol{x}_i - \text{diag}(\boldsymbol{b} \circ \boldsymbol{\lambda}) \boldsymbol{A}_j^\top \sum_{i:z(i)=j} (y_i - \hat{\boldsymbol{\mu}}_j) \right)
\end{aligned}$$

where $\hat{\boldsymbol{\mu}}_{\boldsymbol{j}} \triangleq \boldsymbol{\mu}_j + \boldsymbol{A}_j \text{diag}(\boldsymbol{b} \circ \boldsymbol{\lambda}) \boldsymbol{x}_i$

- Sample $\boldsymbol{\Sigma}_j = \text{diag}(\sigma_{j1}^{-1}, \sigma_{j2}^{-1}, \ldots, \sigma_{jK}^{-1})$ from

$$p(\sigma_{jk}|-) = \text{Gamma}(\sigma_{jk}; e_0 + \frac{1}{2} \sum_{i:z(i)=j} 1), f_0 + \frac{1}{2} \sum_{i:z(i)=j} \|x_{ik} - \hat{\mu}_{jk}\|_2^2) \tag{B.11}$$

B.2 Update Equations for Multi-task Dynamic Model

The Gibbs sampling inference algorithm for the multi-task dynamic model can be derived as follows:

- Sample the cluster index $z_{l,t}$ from

$$p(z_{l,t} = h|-) \propto \pi_{lh} \mathcal{N}(\boldsymbol{x}_t^{(l)}; \boldsymbol{m}_{lh}, \boldsymbol{\Sigma}_{lh}^{-1}) \mathcal{N}(\boldsymbol{x}_{t+1}^{(l)}; \boldsymbol{M}_{c_{lh}} \boldsymbol{x}_t^{(l)}, \gamma^{-1} \boldsymbol{I}) \tag{B.12}$$

After normalization across z, $p(z_{l,t} = h|-)$ becomes a Multinomial distribution.

- Sample the local DP concentration parameter α from

$$p(\alpha|-) = \text{Gamma}\left(\alpha; \tau_{10} + L(H-1), \tau_{20} - \sum_{l=1}^{L} \sum_{h=1}^{H-1} \log(1 - V_{lh}) \right) \tag{B.13}$$

- Sample the local DP stick weight V_{lh}, $h = 1, 2, \cdots, H-1$ and $V_{lH} = 1$ from

$$p(V_{lh}|-) = \text{Beta}\left(V_{lh}; 1 + \sum_{z_{l,t}=h} 1, \alpha + \sum_{z_{l,t}>h} 1\right) \tag{B.14}$$

- Sample the cluster index c_{lh} from

$$p(c_{lh} = j|-) \propto \eta_j \mathcal{N}(\boldsymbol{x}_{t+1}^{(l)}; \boldsymbol{M}_j \boldsymbol{x}_t^{(l)}, \gamma^{-1}\boldsymbol{I}) \tag{B.15}$$

  After normalization across c, $p(c_{lh} = j|-)$ becomes a Multinomial distribution.

- Sample the global DP concentration parameter β from

$$p(\beta|-) = \text{Gamma}\left(\beta; \tau_{30} + J^{dyn} - 1, \tau_{40} - \sum_{j=1}^{J^{dyn}-1} \log(1 - U_j)\right) \tag{B.16}$$

- Sample the global DP stick weight U_j, $j = 1, 2, \cdots, J^{dyn} - 1$ and $U_{J^{dyn}} = 1$ from

$$p(U_j|-) = \text{Beta}\left(U_j; 1 + \sum_{c_{l,z_{l,t}}=j} 1, \alpha + \sum_{c_{l,z_{l,t}}>j} 1\right) \tag{B.17}$$

- Sample λ_k from

$$p(\lambda_k|-) = \text{Gamma}(a_0 + \frac{KJ^{dyn}}{2}, b_0 + \frac{1}{2}\sum_{d=1}^{K}\sum_{j=1}^{J^{dyn}} M_{jkd}^2) \tag{B.18}$$

  where M_{jkd} is the d-th element of $\boldsymbol{M}_{jk}$

- Sample $\boldsymbol{M}_{jk}$ from

$$p(\boldsymbol{M}_{jk}|-) = \mathcal{N}(\boldsymbol{\mu}_{jk}, \boldsymbol{\Lambda}_{jk}) \tag{B.19}$$

$$\begin{aligned}
\boldsymbol{\Lambda}_{jk} &= \left(\gamma \sum_{c_{l,z_{l,t}}=j} {x_{tk}^{(l)}}^2 \boldsymbol{I} + \mathrm{diag}(\boldsymbol{\lambda}) \right)^{-1} \\
\boldsymbol{\mu}_{jk} &= \boldsymbol{\Lambda}_{jk} \left(\gamma \sum_{c_{l,z_{l,t}}=j} x_{tk}^{(l)} (x_{t+1}^{(l)} - \sum_{i \neq k} \boldsymbol{M}_{ji} x_{ti}^{(l)}) \right)
\end{aligned}$$

- Sample the mean vector and covariance matrix for each cluster $\boldsymbol{m}_{lh}, \boldsymbol{\Sigma}_{lh}$ from

$$\nu_{lh} = \nu_0 + \sum_{z_{l,t}=h} 1, \quad u_{lh} = u_0 + \sum_{z_{l,t}=h} 1, \quad \boldsymbol{m}_{lh} = \frac{u_0 \boldsymbol{m}_0 + \sum_{z_{l,t}=h} \boldsymbol{x}_t^{(l)}}{u_{lh}},$$

$$\begin{aligned}
\boldsymbol{\Sigma}_{lh}^{-1} &= \boldsymbol{B}_0^{-1} + \sum_{z_{l,t}=h} \boldsymbol{x}_t^{(l)} {\boldsymbol{x}_t^{(l)}}^T + u_0 \boldsymbol{m}_0 \boldsymbol{m}_0^T - \\
&\qquad \frac{(u_0 m_0 + \sum_{z_{l,t}=h} \boldsymbol{x}_t^{(l)})(u_0 m_0 + \sum_{z_{l,t}=h} \boldsymbol{x}_t^{(l)})^T}{u_{lh}}
\end{aligned} \tag{B.20}$$

- Sample the precision of the additive noise γ_l from

$$p(\gamma_l | -) = \mathrm{Gamma}(\gamma_l; e_l, f_l) \tag{B.21}$$

with

$$e_l = e_0 + \frac{K(T_l - 1)}{2}; \quad f_l = f_0 + \frac{1}{2} \sum_{t=1}^{T_l - 1} \| \boldsymbol{x}_{t+1}^l - \boldsymbol{M}_j \boldsymbol{x}_t^l \|_2^2$$

Appendix C

Summary of Update Equations for HMT Model

C.1 Algorithm 1. Retrospective Inference for c_{di}

Give the initialization $c_{di}, d \in D, i \in N_d$, and set $S = \max\{c_{di}\}$

Step 1 Sample $\boldsymbol{\phi}_{st}$ from its conditional posterior, $s \leq S$

Step 2.1 Sample ν_s from its conditional posterior, $s \leq S$

Step 2.2 Calculate $\pi_s = \nu_s \prod_{h=1}^{s-1}(1-\nu_h)$, $s \leq S$

Step 3.1 Repeat the following until $d > D$ and $i > N_d$

Step 3.2 Sample $u_{di} \sim \text{Uniform}(0,1)$

Step 3.3.1 if $\sum_{l=0}^{s'-1} q_{di}(l) < u_{di} < \sum_{l=1}^{s'} q_{di}(l)$ is true for some $s' \leq S$, then set $c_{di} = s'$ with probability $\kappa_{di}(s, s')$, otherwise, leave c_{di} unchanged. If $i < N_d$, set $i = i + 1$, else set $d = d + 1$ and $i = 1$. Finally goto Step 3.1.

Step 3.3.2 if $u_{di} > \sum_{l=1}^{S} q_{di}(l)$, set $S = S{+}1, c_{di} = S$. Sample $\{\nu_{dS}\}, \{\boldsymbol{\phi}_{St}\}, z_{di}^{S}, P^{(S-1)}$,$\boldsymbol{b}_{S-1}$ from the prior and set $\pi_{dS} = \nu_{dS} \prod_{h=1}^{S-1}(1-\nu_{dh})$, $\boldsymbol{b}_S = 1 - \sum_{h=1}^{S-1} \boldsymbol{b}_s$, goto Step 3.3.1

C.2 Update Equations for Remaining Variables in HMT model

Define $A_s \triangleq \{v : b_{sv} = 1, v \in \mathcal{V}\}$ to be the set of indices of $\boldsymbol{b}_s$ that are utilized. Let $n_{st}^{(v)}$ denote the number of times that term v has been assigned to topic t in scale s, and let $n_{st}^{(\cdot)}$ denote the number of times that all the terms have been assigned to topic t in scale s.

The Gibbs sampling inference procedure is described as follow:

- Sampling transition matrix $\mathbf{P}^{(s)}$:

$$V_{tm}^{(s)} \sim \text{Beta}(1 + \sum_{d,i} 1(z_{di}^{(s+1)} = t, z_{di}^{(s)} = m), \zeta_s + \sum_{d,i} 1(z_{di}^{(s+1)} > t, z_{di}^{(s)} = m)) \tag{C.1}$$

$$p_{1m}^{(s)} = V_{1m}^{(s)}, \quad p_{tm}^{(s)} = V_{tm}^{(s)} \prod_{l=1}^{t-1} (1 - V_{lm}^{(s)}) \tag{C.2}$$

  For new values, set $p_{T_{s+1}+1,m}^{(s)} = 1 - \sum_{l=1}^{T_{s+1}} p_{l,m}^{(s)}$ and draw a new column for $\mathbf{P}^{(s+1)}$ from the prior. More details can be found in [PC09a].

- Sampling term scale indicators $\boldsymbol{b}_s$:

$$\begin{aligned} p(\boldsymbol{b}_s|-) &\propto p(\boldsymbol{b}_s|\tau_s, \boldsymbol{b}_{[s-1]}) \prod_{(d,i):c_{di}=s} p(w_{di}|\boldsymbol{b}_s) \\ &= p(\boldsymbol{b}_s|\tau_s, \boldsymbol{b}_{[s-1]}) \int d\boldsymbol{\phi}_{st} \{p(\boldsymbol{\phi}_{st}|\boldsymbol{b}_s) \prod_{(d,i):c_{di}=s} p(w_{di}|\boldsymbol{\phi}_{st})\} \\ &= p(\boldsymbol{b}_s|\tau_s, \boldsymbol{b}_{[s-1]}) \prod_{t=1}^{T_s} \frac{\Gamma(\gamma|A_s|) \prod_{v \in A_s} \Gamma(n_{st}^{(v)} + \gamma)}{\Gamma^{|A_s|}(\gamma) \Gamma(n_{st}^{(\cdot)} + \gamma|A_s|)} \end{aligned} \tag{C.3}$$

  where $|A_s| = \sum_v b_{sv}$ and $\Gamma(\cdot)$ is the gamma function. In addition, $\boldsymbol{b}_S = 1 - \sum_{h=1}^{S-1} \boldsymbol{b}_s$.

- Sampling Bernoulli parameter τ_s

$$\tau_s \sim \text{Beta}(1 + |A_s|, \psi + \sum_{l=s+1}^{S} |A_l|) \tag{C.4}$$

- Sampling topic indicator $z_{di}^{(s)}$:

$$p(z_{di}^{(s)}|-) \propto p(z_{di}^{(s)}|\boldsymbol{\theta}_d)[p(w_{di}|\boldsymbol{\phi}_{1,z_{di}^{(1)}})1(c_{di}=1) + p(z_{di}^{(2)}|z_{di}^{(1)})1(c_{di}>1)] \tag{C.5}$$

After normalization, $p(z_{di}^{(s)}|-)$ becomes a multinomial distribution.

- Sampling stick length $\alpha_t, t = 1, \ldots, T_1$

$$\alpha_t = \beta_t \prod_{l=1}^{t-1}(1-\beta_l), \beta_t \sim \text{Beta}(1 + m_{\cdot t}, \lambda + \sum_{l>t} m_{\cdot l}) \tag{C.6}$$

where $\boldsymbol{m}$ is an auxiliary table count random variable and $m_{\cdot t}$ denotes the number of tables serving dish t [TJBB06].

- Sampling distribution over topics $\boldsymbol{\theta}_d$

$$\theta_{dt} = \tilde{\theta}_{dt} \prod_{l=1}^{t-1}(1-\tilde{\theta}_{dl}) \tag{C.7}$$

$$\tilde{\theta}_{dt} \sim \text{Beta}(\eta\beta_t + \sum_i 1(z_{di}^{(1)}=t), \eta(1 - \sum_{l=1}^{t}\beta_l) + \sum_i 1(z_{di}^{(1)}>t)) \tag{C.8}$$

- Sampling topic distributions $\boldsymbol{\phi}_{st}$

$$\boldsymbol{\phi}_{st} \sim \text{Dirichlet}(\boldsymbol{\kappa}_{st}) \tag{C.9}$$

where $\kappa_{stv} = \gamma b_{sv} + \sum_d \sum_i 1(c_{di}=s, z_{di}^{(s)}=t, w_{di}=v)$

- Sampling procedure for table count $\boldsymbol{m}$ is exactly the same as that in the HDP [TJBB06]. In addition, Gamma prior is placed on ψ, λ, ρ, ζ_s and η. The sampling procedure can also be referred to HDP. we omit the details here.

Bibliography

[Ach03] D. Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal Comp. Syst. Sc.*, 66:2003, 2003.

[AEB06] M. Aharon, M. Elad, and A. M. Bruckstein. The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representations. *IEEE Transaction Signal Processing*, 54, 2006.

[AGJ10] R. P. Adams, Z. Ghahramani, and M. I. Jordan. Tree-structured stick breaking for hierarchical data. In *NIPS*, 2010.

[Bar07] R. G. Baraniuk. Compressive sensing. *IEEE Signal Processing Magazine*, 24, 2007.

[BD09] J. L. Bigelow and D. B. Dunson. Bayesian semiparametric joint models for functional predictors. *Journal of the American Statistical Association*, 104:26–36, 2009.

[BDE07] A. M. Bruckstein, D. L. Donoho, and M. Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM review*, 51:34–81, 2007.

[Bea03] M. J. Beal. *Variational algorithms for approximate Bayesian inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.

[BGJ10] D. M. Blei, T. L. Griffiths, and M. I. Jordan. The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies. *Journal ACM*, 57(2), 2010.

[BGJT03] D. M. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. In *NIPS*, 2003.

[BNJ03] D. M. Blei, A. Ng, and M. I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[CDS99] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20:33–61, 1999.

[CL06] R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis: Special issue on Diffusion Maps and Wavelets*, 21(1):5–30, 2006.

[CSP+10] M. Chen, J. Silva, J. Paisley, C. Wang, D. B. Dunson, and L. Carin. Compressive sensing on manifolds using a nonparametric mixture of factor analyzers: Algorithm and performance bounds. *IEEE Transaction on Signal Processing*, 12:6140–6156, 2010.

[CT06] E. Candès and T. Tao. Near-optimal signal recovery from random projections and universal encoding strategies. *IEEE Transaction on Information Theory*, 52, 2006.

[DCS09] J. M. Duarte-Carvajalino and G. Sapiro. Learning to sense sparse signals: Simultaneous sensing matrix and sparsifying dictionary optimization. *IEEE Transaction on Image Processing*, 18(7):1395–1408, 2009.

[DDT+08] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 2008.

[DM05] P. Drineas and M. W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, 6, 2005.

[Don06] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52:1289–1306, 2006.

[DTDS06] D. L. Donoho, Y. Tsaig, I. Drori, and J. Starck. Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit. *Stanford Statistics Technical Report 2006-2*, April 2006.

[EA06] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transaction on Image Processing*, 15, 2006.

[EHJT04] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics (with discussion)*, 32:407–499, 2004.

[EM09] Y. C. Eldar and M. Mishali. Robust recovery of signals from a structured union of subspaces. *IEEE Transaction Information Theory*, 55:5302–5316, 2009.

[EY10] M. Elad and I. Yavneh. A weighted average of sparse representations is better than the sparsest one alone. *Preprint*, 2010.

[Fer73] T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1(2):209–230, 1973.

[FSJW10] E. B. Fox, E. B. Sudderth, M. I. Jordan, and A. S. Willsky. Sharing features among dynamical systems with Beta processes. In *NIPS*, 2010.

[GB00] Z. Ghahramani and M. J. Beal. Variational inference for Bayesian mixtures of factor analysers. In *NIPS*, 2000.

[GE10] S. Gleichman and Y. C. Eldar. Blind compressed sensing. *Preprint (on Arxiv. org)*, 2010.

[GG05] T. L. Griffiths and Z. Ghahramani. Infinite latent feature models and the Indian buffet process. In *Advances in Neural Information Processing Systems*, pages 475–482, 2005.

[GMHP04] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popovic. Style-based inverse kinematics. In *Proc. SIGGRAPH*, 2004.

[GST07] T. L. Griffiths, M. Steyvers, and J. B. Tenenbaum. Topics in semantic representation. *Psychological Review*, 114(2):211–244, 2007.

[HDR97] G. E. Hinton, P. Dayan, and M. Revow. Modeling the manifolds of images of handwritten digits. *IEEE Transaction on Neural Networks*, 8:65–74, 1997.

[HLMS04] J. Ham, D. D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. In *Proc. International Conference on Machine Learning*, 2004.

[Hof99] T. Hofmann. The cluster-abstraction model: Unsupervised learning of topic hierarchies from text data. In *IJCAI*, pages 682–687, 1999.

[Hof09] P. D. Hoff. Simulation of the matrix Bingham-von Mises-Fisher distribution, with applications to multivariate and relational data. *Journal Comp. Graph. Statistics*, 2009.

[HZM09] J. Huang, T. Zhang, and D. Metaxas. Learning with structured sparsity. *ICML*, 2009.

[IJ01] H. Ishwaran and L. F. James. Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, 2001.

[JL82] W. B. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Conf. Modern Anal. Probability*, pages 189–206, 1982.

[JMOB10] R. Jenatton, J. Mairal, G. Obozinski, and F. Bach. Proximal methods for sparse hierarchical dictionary learning. In *ICML*, Haifa, Israel, 2010.

[JXC08] S. Ji, Y. Xue, and L. Carin. Bayesian compressive sensing. *IEEE Transactions on Signal Processing*, 56:2346–2356, 2008.

[KG07] D. Knowles and Z. Ghahramani. Infinite sparse factor analysis and infinite independent components analysis. In *7th International Conference on Independent Component Analysis and Signal Separation*, 2007.

[KWX+04] B. Krishnapuram, D. Williams, Y. Xue, A. J. Hartemink, L. Carin, and M. A. T. Figueiredo. On semi-supervised classification. In *NIPS*, 2004.

[Law05] N. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 2005.

[LB04] E. Levina and P. Bickel. Max. likelihood estimation of intrinsic dimension. In *NIPS*, 2004.

[LM07] N. D. Lawrence and A. J. Moore. Hierarchical Gaussian process latent variable models. In *Proc. International Conference on Machine Learning*, 2007.

[MBP+08] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Supervised dictionary learning. In *Proceedings of Neural and Information Processing Systems*, 2008.

[MBP+09] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Non-local sparse models for image restoration. In *Proc. International Conference on Computer Vision*, 2009.

[MBPS09] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the International Conference on Machine Learning*, 2009.

[MES08] J. Mairal, M. Elad, and G. Sapiro. Sparse representation for color image restoration. *IEEE Transaction on Image Processing*, 17:53–69, 2008.

[MSE08] J. Mairal, G. Sapiro, , and M. Elad. Learning multiscale sparse representations for image and video restoration. *SIAM Multiscale Modeling and Simulation*, 7:214 – 241, 2008.

[PC09a] J. Paisley and L. Carin. Hidden Markov models with stick-breaking priors. *IEEE Transaction on Signal Processing*, 57(10):3905–3917, 2009.

[PC09b] J. Paisley and L. Carin. Nonparametric factor analysis with Beta process priors. In *Proc. International Conference on Machine Learning*, 2009.

[PR08] O. Papaspiliopoulos and G. O. Roberts. Retrospective Markov chain Monte Carlo methods for Dirichlet process hierarchiacal models. *Biometrika*, 95(1):169–186, 2008.

[RBL+07] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *Proc. International Conference on Machine Learning*, 2007.

[RD09] A. Rodriguez and D. B. Dunson. Nonparametric Bayesian models through probit stickbreaking processes. *University California Santa Cruz Technical Report*, 2009.

[RDC11] L. Ren, L. Du, and L. Carin. The logistic stick breaking process. *Journal Machine Learning Research*, 12:203–239, 2011.

[RPCL06] M. Ranzato, C. Poultney, S. Chopra, and Y. Lecun. Efficient learning of sparse representations with an energy-based model. In *NIPS*, 2006.

[RS00] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 2000.

[RS10] I. Ramirez and G. Sapiro. Universal sparse modeling. *arXiv:1003. 2941*, 2010.

[Set94] J. Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4:639–650, 1994.

[SK96] M. Smith and R. Kohn. Nonparametric regression using Bayesian variable selection. *Journal of Econometrics*, 75:317–344, 1996.

[SPB10] M. Shankar, N. P. Pitsianis, and D. J. Brady. Compressive video sensors using multichannel imagers. *Appl. Opt.*, 49, 2010.

[SSM98] B. Schölkopf, A. Smola, and K. R. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 1998.

[TB99a] M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11:443–482, 1999.

[TB99b] M. E. Tipping and C. M. Bishop. Probabistic principal component analysis. *Journal of the Royal Statistical Society B*, pages 611–622, 1999.

[TdSL00] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 2000.

[TG07] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53:4655–4666, 2007.

[THR07] G. W. Taylor, G. E. Hinton, and S. T. Roweis. Modeling human motion using binary latent variables. In *NIPS*, 2007.

[Tip01] M. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1, June 2001.

[TJ07] R. Thibaux and M. I. Jordan. Hierarchical beta processes and the Indian buffet process. In *Proc. International Conference on Artificial Intelligence and Statistics*, 2007.

[TJBB06] Y. W. Teh, M. I. Jordan, Matthew J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[TP91] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3:71–83, 1991.

[TR03] Y. Teh and S. Roweis. Automatic alignment of local representations. In *NIPS*, 2003.

[Ver06] J. Verbeek. Learning nonlinear image manifolds by global alignment of local linear models. *IEEE Transaction Pattern Analysis Machine Intelligence*, 2006.

[VRV04] J. J. Verbeck, S. T. Roweis, and N. Vlassis. Nonlinear CCA and PCA by alignment of local modes. In *NIPS*, 2004.

[Wak10] M. B. Wakin. Manifold-based signal recovery and parameter estimation from compressive measurements. *arXiv:1002. 1247v1*, 2010.

[WB09a] C. Wang and D. M. Blei. Decoupling sparsity and smoothness in the discrete hierarchical Dirichlet process. In *NIPS*, 2009.

[WB09b] C. Wang and D. M. Blei. Variational inference for the nested Chinese restaurant process. In *NIPS*, 2009.

[Wes03] M. West. Bayesian factor regression models in the "large p, small n" paradigm. *Bayesian Statistics*, 7:723–732, 2003.

[WFH08] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transaction Pattern Analysis Machine Intelligence*, 2008.

[WWHB10] S. Williamson, C. Wang, K. A. Heller, and D. M. Blei. The IBP compound Dirichlet process and its application to focused topic modeling. In *ICML*, Haifa, Israel, 2010.

[WYG+09] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Transaction Pattern Analysis Machine Intelligence*, 2009.

[ZCP+09] M. Zhou, H. Chen, J. Paisley, L. Ren, G. Sapiro, and L. Carin. Non-parametric Bayesian dictionary learning for sparse image representations. In *Proc. Neural Information Processing Systems*, 2009.

[ZMP04] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *NIPS*, 2004.

[ZZ04] Z. Zhang and H. Zha. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM Journal Sci. Comput.*, 2004.

Biography

Haojun Chen was born in December, 1979 in Guangzhou, China. He received his B.S. in Electronic Engineering and M.S. degree in Signal and Information Processing from Xidian University, Xi'an, China. He began to study in the Department of Electrical and Computer Engineering at Duke University, Durham, NC, where he received a M.S. in Electrical and Computer Engineering in 2009. He is now pursuing a PhD degree in the Department of Electrical and Computer Engineering at Duke University. His current research interests include Bayesian statistics and machine learning, with focus on compressive sensing, manifold learning and topic modeling.

Publications

1. H. Chen, J. Silva, D. Dunson and L. Carin, Hierarchical Bayesian Embeddings for Analysis and Synthesis of Dynamic Data, submitted to *IEEE Trans. Signal Processing*

2. L. He, H. Chen and L. Carin, Tree-Structured Compressive Sensing with Variational Bayesian Analysis, *IEEE Signal Processing Letter*, Vol. 17 (3), 2010

3. M. Zhou, H. Chen, J. Paisley, L. Ren and L. Carin, Non-Parametric Bayeisan Dictionary Learning for Sparse Image Representations, *In Proceedings of the Neural Information Processing Systems (NIPS)* Vancouver, Canada, 2009.

4. M. Zhou, H. Chen, J. Paisley, L. Ren, L. Li, Z. Xing, D. Dunson, G. Sapiro and

L. Carin, Nonparametric Bayesian Dictionary Learning for Analysis of Noisy and Incomplete Images, submitted to *IEEE Trans. Image Processing*

5. S. D. Feller, H. Chen, D. J. Brady, M. E. Gehm, C. Hsieh, O. Momtahan and A. Adibi, Multiple order coded aperture spectrometer, *Optics Express*, 15(9): 5625-5630, 2007.