SPECTRAL ESTIMATION OF HIDDEN MARKOV MODELS

Jordan Rodu

A DISSERTATION

in

Statistics

For the Graduate Group in
Managerial Science and Applied Economics

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2014

Supervisor of Dissertation

_____

Dean Foster
Marie and Joseph Melone Professor
of Statistics

Graduate Group Chairperson

_____

Eric Bradlow, The K.P. Chao Professor of
Marketing, Statistics, and Education

Dissertation Committee

Dean Foster, Professor of Statistics
Dylan Small, Professor of Statistics
Robert A Stine, Professor of Statistics
Lyle Ungar, Professor of Computer Science
Abraham J Wyner, Professor of Statistics

SPECTRAL ESTIMATION OF HIDDEN MARKOV MODELS

Jordan Rodu

Acknowledgments

I would like to thank my advisor Dean Foster from whom I have learned so much in the course of five short years. I have extremely enjoyed working with you and look forward to future collaborations.

Many thanks also go to Lyle Ungar who has helped me immensely, especially in teaching me how to effectively present our research, both in paper form and in talks. Many thanks go to the rest of my committee, Dylan Small, Robert Stine (to whom I am particularly indebted for his edits and suggestions on this document), and Abraham Wyner for excellent discussion and general research advice.

Special thanks go to Shane Jensen and Dylan Small with whom I have extremely enjoyed collaborating on biologically oriented applications. I am excited to continue to work with both of you for many years to come.

Thanks to Adam Kapelner for making this beautiful dissertation template, and to he and Alex Goldstein for hours of discussions of just about everything imagineable over the years.

Mike "my boo" Baiocchi has been my best friend throughout my entire PhD and has provided endless advice and guidance. I look forward to more-than-annual visits, and hopefully one day finding our common collaboration ground.

Thanks to Lindsey Fiorelli for making this year better than I could have possibly imagined a fifth year being, and for making me laugh harder, and longer, than I have ever laughed.

Lastly, thanks to my family. Thanks to my dad for instilling in me academic values, and for teaching me how to roast coffee. Thanks to my mom for doing all of my worrying for me, and for being so understanding when I failed to call. Hopefully the tenure clock will be starting soon... just saying...

ABSTRACT

SPECTRAL ESTIMATION OF HIDDEN MARKOV MODELS

Jordan Rodu

Dean Foster

This thesis extends and improves methods for estimating key quantities of hidden Markov models through spectral method-of-moments estimation. Unlike traditional estimation methods like EM and Gibbs sampling, the set of estimation methods, which we call spectral HMMs (sHMMs), are incredibly fast, do not require multiple restarts, and come with provable guarantees. Our first result improves upon the original spectral estimation of hidden Markov models algorithm by estimating the parameters from fully reduced data. We also show that the parameters developed in the fully reduced dimensional version can be estimated using various forms of regression, which can lead to major speed gains, as well as allowing flexibility in the estimation scheme. We then extend the algorithm beyond basic hidden Markov models to latent variable tree structures that have linguistic applications, especially dependency parsing, and finally to hidden Markov models in which the output is a high-dimensional, continuously distributed variable. We show that spectral estimation of hidden Markov models can be factored into two major components- estimation of the hidden state space dynamics, and estimation of the observation probability distributions. This leads to extremely flexible estimation procedures that can be tailored precisely for the task of interest. These tools are all simple to implement, fast, and naturally incorporate dimension reduction, which allows them to scale gracefully as the dimension of the data increases.

Contents

List of Tables

List of Figures

1 Introduction

Hidden Markov Models (HMMs) are a class of latent state models that are widely used in many domains. HMMs are useful for stochastic data, and can be used with both discrete and continuous data. There are two primary uses of HMMs: identifying the likelihood of a sequence of observations given a model, and generating features.

Obtaining the likelihood of a sequence of observations from a model is useful for classification tasks. An interesting application of HMMs is in gene finding. Essentially an HMM is trained on a sequence of base pairs known to be a gene, and then an unknown sequence of base pairs is fed into the model. If the probability of this sequence is high, then the sequence is labeled as a gene, otherwise it is not (see, for instance, Huang et al. (1990)).

Besides the likelihood of a sequence of observations, HMMs can also generate at each time step a vector indicating the probability of being in each hidden state at that time given the history of observations. This belief vector can be used, for instance, as a feature for some classification algorithm. Robot localization is one domain that relies on these estimates. The robot is given the task to track its location inside a building given some measurement. These measurements can range from visual data collected by the robot, to signal data- like signal strength from wifi routers. Of course, images or signals are almost never unique, and are often quite noisy. Hence, localization from this data alone is not possible. Estimates can be quite improved if the robot uses a belief vector given previous observations. Intuitively, while two observations may look extremely similar, one of those observations might be highly unlikely given the current state of the robot (established from previous observations). For more on robot localization, see for instance Thrun et al. (1998).

Beyond gene recognition and robot localization, HMMs are a part of the state–of–the–art toolkits of many domains, including speech recognition (continuous data, and among the oldest, most successful applications of HMMs), gesture recognition (also continuous data), and natural language processing (NLP) tasks (often discrete data like words, and another major success story for HMMs).

Recent years have seen an explosion of data, both in terms of quantity and dimensionality. The internet now houses billions of webpages. 100 hours of video are uploaded to Youtube every minute. Financial trading data on thousands of stocks

are available at frequencies of fractions of a second. There is a major need for fast, efficient tools to estimate the key quantities of HMMs– ways that can handle such high–dimensional data. Spectral methods for estimating HMMs provide that platform.

In this chapter we will introduce more formally the HMM, review spectral method of moment estimation in a way that extracts the intuition behind the algorithms, and motivate the material in the remainder of the dissertation.

1.1 The Hidden Markov Model

The basic HMM is a generative model that consists of a chain of latent states that, at each time point, emit observations from a distribution that depends on the current hidden state (see figure 1.1 for a graphical representation). There are two primary assumptions for this basic HMM:

(a) The underlying hidden state process is Markovian

(b) Given the hidden states, the observations are independent

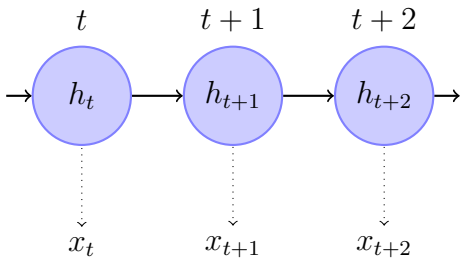

Figure 1.1: HMM with states h_t, h_{t+1}, and h_{t+2} which emit observations x_t, x_{t+1}, and x_{t+2} respectively.

For the hidden state sequence to be Markovian means that, at time t, the probability distribution over the next hidden state at time $t+1$ depends only on the value of the current hidden state at time t, so

$$\Pr(h_{t+1}|h_t, \ldots, h_1) = \Pr(h_{t+1}|h_t)$$

We can fully specify the basic HMM with a probability distribution over the initial hidden state, often denoted by π, a parameter capturing the probability of transition from hidden state to hidden state, encapsulated in a matrix T, and a parameter that indicates the probability of a particular emission x given the current hidden state.

$$[\pi]_i = \Pr(h_1 = i)$$

$$[T]_{i,j} = \Pr(h_{t+1} = i | h_t = j)$$
$$[\lambda(x)]_i = \Pr(x_t | h_t = i)$$

In this dissertation we will concern ourselves with a particular type of HMM– one in which the hidden state space, of dimension k, is much smaller than the dimension of the observation space, v. Further, the basic HMM can be generalized to include a state space that is not discrete or to the continuous time setting. We will not be considering these extensions in this dissertation, however we will consider both a discrete and continuous observation space.

For completeness, the likelihood of a sequence of observations is

$$P(x_1, \ldots, x_t) = \sum_{h_1, \ldots, h_t} [\pi]_{h_1} \prod_{j=2}^{t} [T]_{h_j, h_{j-1}} \prod_{j=1}^{t} [\lambda(x_j)]_{h_j}$$

For fun, let's consider a pictorial view of the parameters of an HMM, seen in figure 1.2.

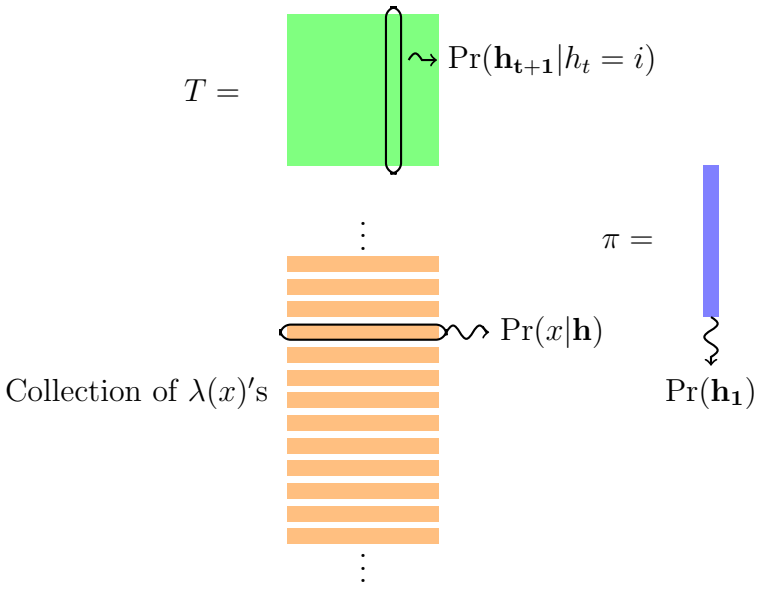

Figure 1.2: Pictorial view of HMM parameters

We can recast the probability statement in terms of a "new" formula,

$$P(x_1, \ldots, x_t) = 1^\top A(x_t) \cdots A(x_1) \pi$$

where $A(x) = T\text{diag}(\lambda(x))$, and diag takes a vector x and puts its entries on the diagonal of a matrix otherwise populated with 0's. For those familiar with forward–backward algorithm, this is the matrix form of the forward calculation. Pictorially, $A(x)$ can be seen in 1.3.

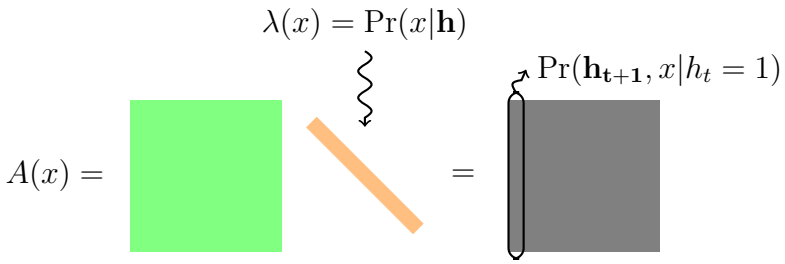

Figure 1.3: $A(x)$, graphically

This formulation of the likelihood is referred to as the "Observable Operator Model" (Jaeger (2000)) in spectral HMM literature. Each observation x is mapped to a matrix $A(x)$, and these matrices are multiplied together to return the probability of the sequence of observations.

Let's look at a concrete example, pictorially. Consider the sentence "Kilroy was here". To calculate the likelihood of this sentence from an HMM with parameters π, T, and λ, we have, from the "forward algorithm" viewpoint we get the picture in figure 1.4

Pr("Kilroy was here") =

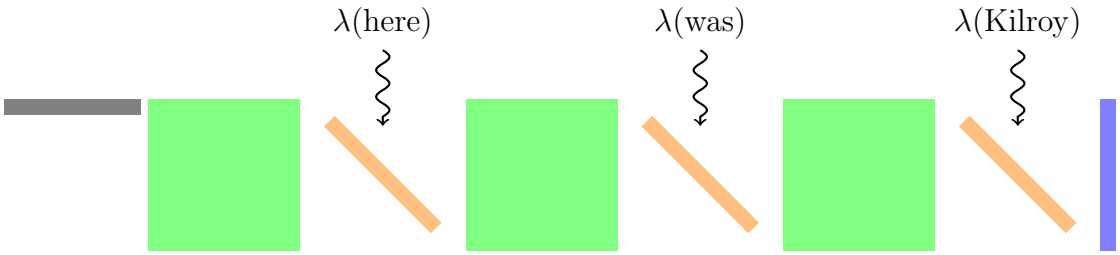

Figure 1.4: "Forward algorithm" version of Pr("Kilroy was here")

This, then, can be represented in operator form as in figure 1.5.

The key to all of this is that, for many quantities of interest, like the probability of a string of observations, recovering T and $\lambda(x)$ isn't necessary. If one can learn an $A(x)$ for each observation, this is sufficient. Unfortunately, $A(x)$ isn't directly learnable. However an appropriate similarity transformation of $A(x)$ (of which there are more than one) is learnable by the method of moments, bypassing the need to recover the HMM parameters, and still gets us what we want. Note that

$$\begin{aligned} P(x_1, \ldots, x_t) &= 1^\top \quad A(x_t) \quad \cdots \quad A(x_1) \quad \pi \\ &= \underbrace{1^\top S^{-1}}_{b_\infty^\top} \quad \underbrace{SA(x_t)S^{-1}}_{B(x_t)} \quad S \quad \cdots \quad S^{-1} \quad SA(x_1)S^{-1} \quad \underbrace{S\pi}_{b_1} \\ &\equiv b_\infty^\top \quad B(x_t) \quad \cdots \quad B(x_1) \quad b_1 \end{aligned}$$

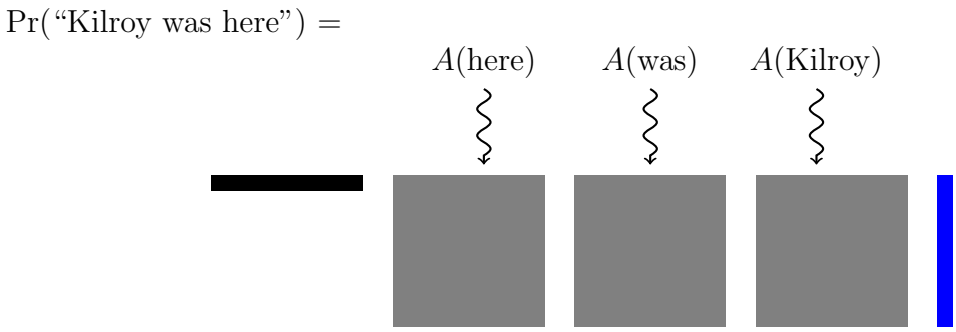

Figure 1.5: "Observable Operator" version of Pr("Kilroy was here")

For now let's just trust that we can learn $b_\infty^\top$ and b_1. For those familiar with the literature, or for those who plan to delve into the literature, this is launching point for Hsu et al. (2009), who show how to learn, for DISCRETE OBSERVATIONS, these truly observable operators $B(x)$ through the method of moments. It requires estimation of the first three moments, as seen in figure 1.6, and an "eigendictionary," U, that maps these moments to low dimensional embeddings. Before proceeding with the story, let's take a quick look at one possible eigendictionary U, and it's function in spectral methods.

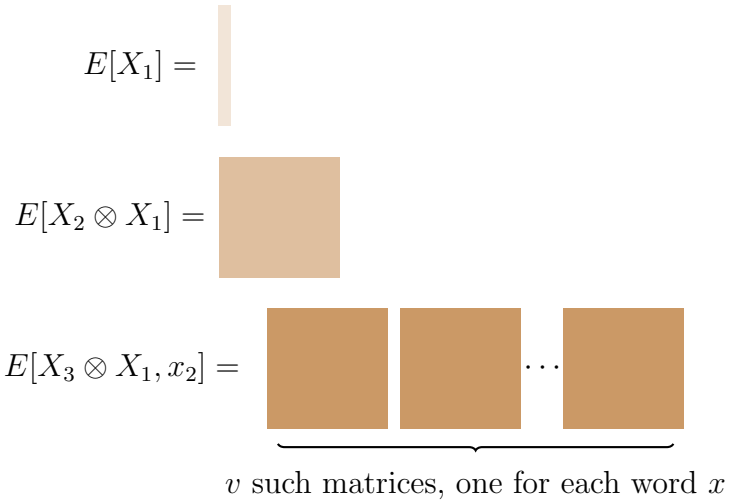

Figure 1.6: Moments needed for estimation of $B(x)$ as in Hsu et al. (2009)

1.2 The Eigendictionary

U maps the three moments (see figure 1.6) into a lower dimensional subspace (of dimension k). An equivalent way to think about U, and indeed the way we think about U in this dissertation, is that it maps *observations* to a lower–dimensional embedding. There are many different eigendictionaries that can work for a given data set. The key is that the mapping cannot lose much of the distributional information about an observation. Recall that our major assumption is that, while observations lie in high dimensional space, they are really governed by a much lower dimensional system.

To get a feel for what an eigendictionary does, let's take a concrete example of words. Let's say our domain has $100,000$ words, and let's assume that our underlying space is about 50 dimensional. In the original space, we map each word to an indicator vector– a vector of length $100,000$ that is all zeros except a single 1 in the entry corresponding to the word of choice.

Each word will be mapped, via the eigendictionary, to a vector of length 50. Now, consider the words "his" and "hers". If we squint, these words are, for all intents and purposes, distributionally equivalent. Wherever we see the word "his", we can almost always substitute "hers". In the original space, these two words are as far apart from each other as they are from any other words. One hope of a good eigendictionary is that these words will be mapped relatively close together– in other words, that their 50–dimensional repreentations will be pointing in pretty much the same direction.

On the other hand, the word "box"– while in the original space is no more distinct from "hers" than "his" is– should be mapped to a low dimensional representation that is pointing in a very different direction than "his" and "hers". Figures 1.7 and 1.8 show examples of the projection of words onto the first and second dimensions of a possible eigendictionary and the projection of words onto the second and third dimensions, respectively.

As mentioned before, there are many possible choices for the eigendictionary U. One possible choice, and the most common from the literature, is to use a subset of the left singular vectors from the singular value decomposition (SVD) of the second moment, $E[X_2 \otimes X_1]$.

1.2.1 A Quick SVD Refresher

The SVD factors a matrix S into the product of two orthonormal matrices U and V, and a diagonal matrix D such that $S = UDV^\top$. This is represented pictorially in the top line of figure 1.9. One aspect of the SVD is that the best rank k approximation of a matrix can be obtained from the first k left singular vectors (corresponding to the first k singular values arranged in descending order), the first k right singular vectors, and the first k singular values. The best rank 1 and 2 matrices are illustrated in the bottom two graphics in figure 1.9.

Let's return to our two words, "his" and "hers". The claim made earlier is that

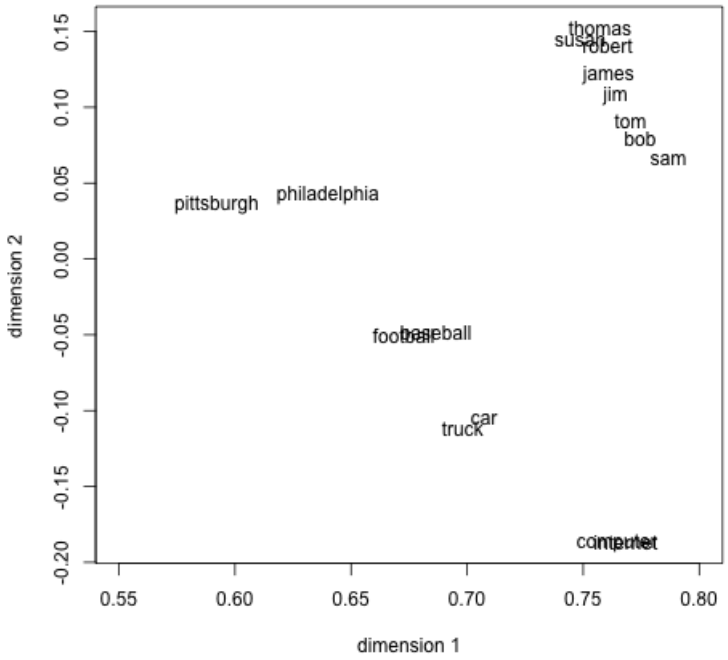

Figure 1.7: Projection of words onto the first two dimensions of the U matrix

these to words look extremely similar, meaning that if we look at the rows corresponding to the words "his" and "hers" in the second moment matrix, these rows should look extremely similar (in other words, "his" and "hers" tend to follow the same words, and co–occur with them in the same proportions). Now, consider the best rank 1 approximation to the second moment matrix, and let's call the first left singular vector u_1 and the first right singular vector v_1 (for now let's ignore the singular values). One way to think about the matrix decomposition is that v_1 is the single best direction that approximates the rows of the second moment matrix, and the entries of u_1 (each of which we can identify with a word in the vocabulary), specify the weight to place on the direction v_1 for a given word. The idea is that the words "his" and "hers" should want almost the same weight for v_1, since they are basically trying to reconstruct the same row, and they should want extremely similar weights for successive right singular vectors. In other words, the weightings given to the words "his" and "hers" in the k left singular vectors should effectively be the same. On the other hand, "box" is trying to reconstruct a very different row than "his" or "hers", and so the weightings of the right singular vectors given to "box" by the left singular vectors should be quite different.

Another thing to note about the SVD is that a matrix of rank k needs exactly k left and right singular vectors, and has k positive, non–zero singular values. The second moment matrix is of size $v \times v$, though because we assume it is of rank k, reconstruction of the theoretical second moment matrix needs only k of the relevant

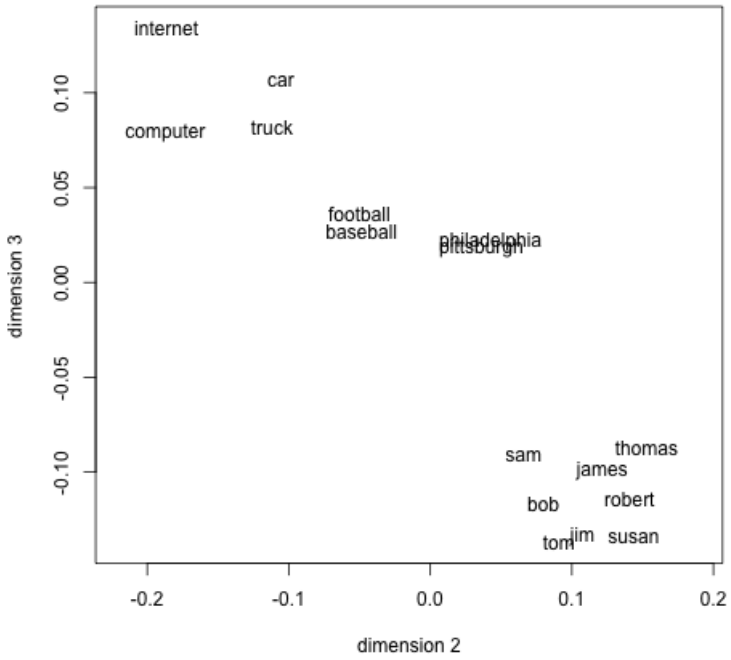

Figure 1.8: Projection of words onto the second two dimensions of the U matrix

components. This is why the eigendictionary, pictured in figure 1.10, needs only k dimensions per observation.

1.3 Returning to the Spectral Story

As mentioned before, Hsu et al. (2009) estimate their observables $B(x)$ using the first three moments of the data. Let's examine a bit more closely the third moment. In their formulation, they estimate v matrices each of size $k \times k$ (figure 1.6). When observation x is observed, the "third moment" matrix corresponding to x is selected for use in building $B(x)$. Alternatively, we can stack them in a tensor, and select the slice of the tensor corresponding to the word of choice (see figure 1.11). Nothing really changes, except now we can think of estimating a single third moment tensor of size $k \times k \times v$, which we can think of as a function that takes a vector (our observation x) and returns a matrix, which can then be used to construct the observable $B(x)$.

One question we can ask, then, is if it is possible to reduce the size of this new third moment "function"– in other words instead of estimating something that is of size $k \times k \times v$, estimating something of size $k \times k \times k$. The answer is yes, and is the subject of chapter 2. To spoil the fun, chapter 2 shows how to estimate the observable operators from the reduced–dimension data (see figure 1.12). We call the observable

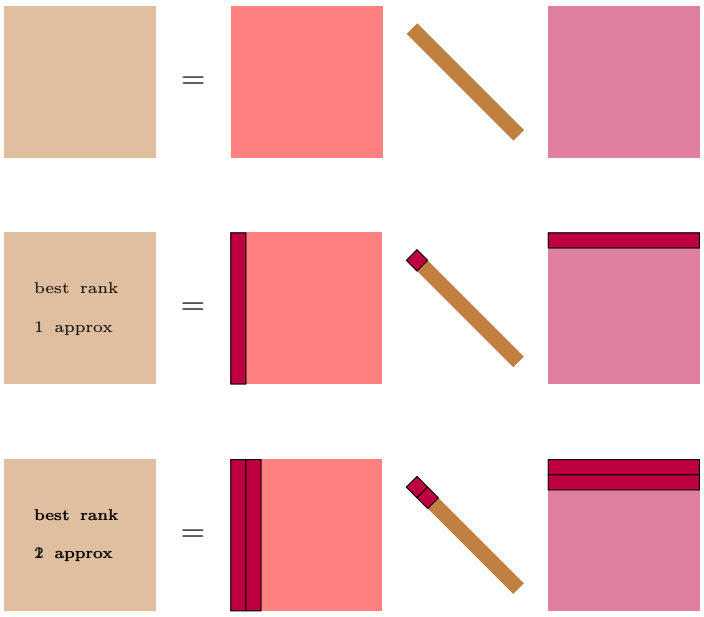

Figure 1.9: The top figure is a pictorial representation of the SVD, the second is the best rank 1 approximation to the matrix, and the third is the best rank 2 approximation

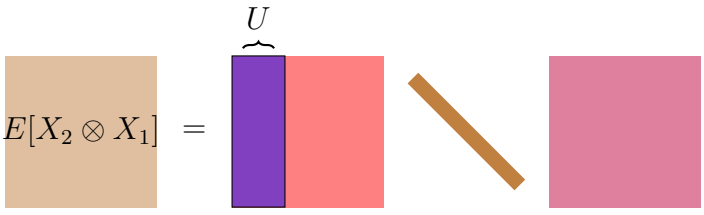

Figure 1.10: The eigendictionary U resulting from the SVD

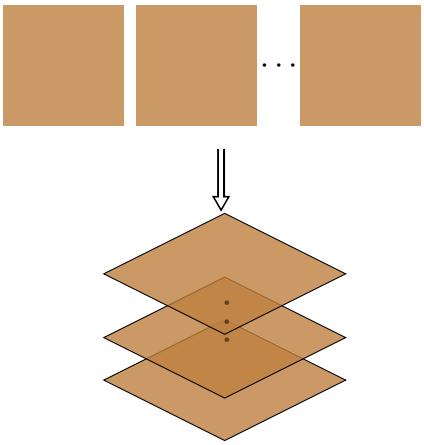

Figure 1.11: Stacking the third moment matrices

operators $C()$ in this case to distinguish them from the $B()$ in Hsu et al. (2009), though for a given observation (in discrete space), $B(x) = C(y)$, where $y = U^\top x$.

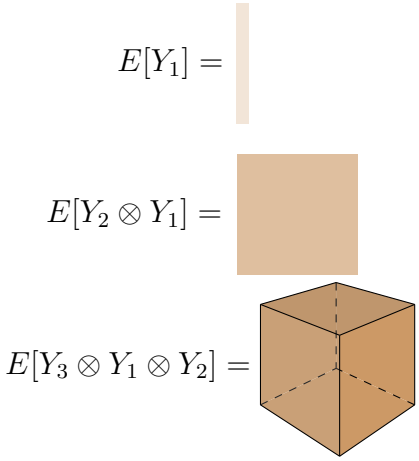

Figure 1.12: Three moments required to build observable operators $C(y)$

Further, chapter 2 provides an analysis of the sample complexity required to estimate the observables $C(y)$.

1.4 Extending Spectral Estimation

So far we have focused on, as does chapter 2, standard HMMs with discrete output. This dissertation extends spectral estimation in two ways. First, chapter 3 extends spectral estimation to latent tree structures. These HMM–like structures, for example in figure 1.13, has the characteristic that the distribution of a hidden node given

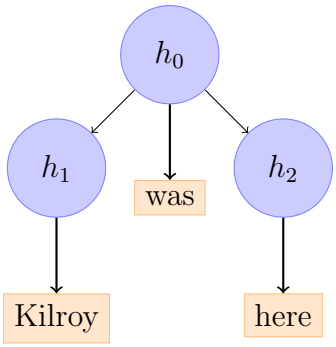

Figure 1.13: Sample dependency parsing tree for "Kilroy was here"

its entire ancestry depends only on its parent node. Further, each node emits an observation.

Chapter 5 extends spectral HMM estimation to HMMs that emit high–dimensional, continuously distributed observations. The key observation in that chapter is that estimation of the observable function $C()$ remains exactly the same as in 2. In fact, spectral estimation of HMMs and HMM–like objects can be factored into two separate parts– estimation of the hidden state space, and estimation of the observation probabilities given the hidden states. The function $C()$ fully encodes all relevant information about the hidden state space. Obtaining observation probabilities depends on the argument passed to the function $C()$. We show in chapter 5 what quantity to estimate in order to isolate the relevant information about the observation space in the continuously distributed observation case.

In particular, one estimates a function $g(x_t) = E[y_{t+1}|x_t]$. While not really the focus of chapter 5, estimation of the function $g(x)$ is extremely flexible, allowing a scientist to choose from a wide array of estimation techniques. Further, one can easily handle missing or incomplete data. This is nice because previous spectral HMM estimation methods were extremely rigid in their estimation procedures, relying exclusively on method of moments to estimate both the hidden state space dynamics and the conditional observation probabilities.

Chapter 4 has a slightly different, though complimentary focus. It focuses on the observable $C()$ and shows that, like $g(x)$, $C()$ can be estimated through various types of regression. This permits a great deal of flexibility, and combined with the estimation of $g(x)$ we now have a factored, extremely flexible way to estimate spectral HMMs. We now have a framework with which to apply spectral estimation to a broader class of latent state models. This dissertation tears apart the estimation problem into two parts– estimation of the hidden state space dynamics and estimation of the observation conditional probabilities– which will hopefully enable researchers to piece together an appropriate estimation procedure in order to estimate their model, further tailoring those estimation procedures to meet the particular characteristics of their data.

1.5 A Word About Notation

A table of notation can be found in appendix A.1. One piece of notation, however, requires a little more attention. We let M be a $v \times k$ matrix where column i of M is the expected value of an observation given the hidden state i. In math, letting M_i represent denote column i of matrix M, we have

$$M_i = E[X|h_i].$$

For spectral methods, M is instrumental as a theoretical quantity in identifying an appropriate similarity transform. In terms of a regular non–spectral HMM estimation procedure, M can also be used to obtain the expected value of an observation given the current hidden state belief vector (so if $\tilde{h}_t$ is the belief probabilities over hidden states at time t, the $Mh_t = E[x_t|\tilde{h}_t]$.

For *discrete* data, M plays another role. When x is an indicator vector (all 0's and a single 1 in the entry corresponding to the given observation), we have the following:

$$E[x|h_i] = \Pr(x|h_i).$$

In other words, for a particular observation x, $\lambda(x)$ can be found simply be extracting the row of M corresponding to observation x:

$$\lambda(x) = M^\top x.$$

For those familiar with HMM literature, M is often written instead as O– for "Observation matrix"– where $[O]_{ij} = \Pr(x = i|h = j)$. One of the goals of this dissertation is to unify estimation in the case where x is discrete and the case where x is continuous. The double loading of O as both an expected value and as a source for $\lambda(x)$ doesn't carry over elegantly to continuous data, and the decision was made to keep the two functions of O separate, hence M and $\lambda(x)$. However, note that in the discrete case $\lambda(x)$ is often simply referred to as $M^\top x$.

2

Reduced Dimensional Estimation of Hidden Markov Models

Hidden Markov Models (HMMs) can be accurately approximated using co-occurrence frequencies of pairs and triples of observations by using a fast spectral method Hsu et al. (2009) in contrast to the usual slow methods like EM or Gibbs sampling. We provide a new spectral method which significantly reduces the number of model parameters that need to be estimated, and generates a sample complexity that does not depend on the size of the observation vocabulary. We present an elementary proof giving bounds on the *relative* accuracy of probability estimates from our model. (Corollaries show our bounds can be weakened to provide either L1 bounds or KL bounds which provide easier direct comparisons to previous work.) Our theorem uses conditions that are checkable from the data, instead of putting conditions on the unobservable Markov transition matrix.

2.1 Introduction

For many applications such as language modeling, it is useful to estimate Hidden Markov Models (HMMs) Rabiner (1989) in which observations drawn from a large vocabulary are generated from a much smaller hidden state. Standard HMM estimation techniques such as Gibbs sampling Geman and Geman (1984) and EM Baum et al. (1970); Dempster et al. (1977) methods, although very widely used, can require some effort to apply as they are often either slow or prone to get stuck in local optima. Hsu, Kakade and Zhang, in a path breaking paper, Hsu et al. (2009) showed that HMMs can, in theory, be efficiently and accurately estimated using closed form calculations on trigrams of observations which have been projected onto a low dimensional space. Key to this approach is the use of singular value decomposition (SVD) on the matrix of covariances between adjacent observations to learn a matrix U that

Work from this chapter appears in Foster et al. (2012)

projects observations onto a space of the same dimension as the hidden state. Perhaps surprisingly, co-occurrence statistics on unigrams, pairs, and triples of observations are sufficient to accurately estimate a model equivalent to the original HMM.

The true hidden state itself cannot, of course, be estimated (it is not observed), but one can estimate a linear transformation of the hidden state which contains sufficient information to give an optimal (in a sense to be made precise below) estimate of the probability of any sequence of observations being generated by the HMM Hsu et al. (2009). The method of Hsu et al. (2009), and the extensions to it presented here do not require EM or Gibbs sampling, but only need an SVD on bigram observation counts. Since SVD is an efficient method guaranteed to return the correct result in a known number of steps, this is a major advantage over the iterative EM method.

Hsu et al. (2009) estimate a size kv matrix mapping between the the dimension v observation space and a reduced dimension space of size k (the dimension of the hidden state space). They also need to estimate a tensor of size vk^2. We provide an alternate formulation that replaces their vk^2 tensor with one of size k^3. Since the observation vocabulary, v, is often much larger than the state space ($v \gg k$), this provides significant reduction in model size, and hence, as we show below, in sample complexity.

2.1.1 HMM set-up and notation

We now introduce the notation and model used throughout this chapter.

Consider an HMM where T is an $k \times k$ transition matrix on the hidden state, M is a $v \times k$ emission matrix giving the probabilities of hidden state $h = j$ emitting observation $x = i$, and π is a vector of initial state probabilities in which π_i is the probability that $h_1 = i$. Jaeger (2000) showed that the joint probability of a sequence of observations from this HMM is given by

$$Pr(x_1, x_2, \ldots, x_t) = 1^\top A(x_t)A(x_{t-1}) \cdots A(x_1)\pi, \tag{2.1}$$

where $A(x) \equiv T\text{diag}(\lambda(x))$, x is the unit vector of length v with a single 1 in the position corresponding to observation x, $\lambda(x)$ is a vector of length k such that $[\lambda(x)]_i = \Pr(x|h = i)$, and $\text{diag}(m)$ creates a matrix with the elements of the vector m on its diagonal and zeros everywhere else. In the case of discrete observations, $\lambda(x) = M^\top x$, and $M^\top x$ will often be used instead of $\lambda(x)$ in this chapter.

$A(x)$ is called an "observation operator", an idea dating back to multiplicity automata (Schutzenbeegeb (1961); Carlyle and Paz (1971); Fliess (1974)), and foundational in the theory of Observable Operator Models (Jaeger (2000)) and Predictive State Representations (Littman et al. (2002)). It is effectively a third order tensor, giving the distribution vector over states at time $t + 1$ as a function of the state distribution vector at the current time t and the current observation x_t. Since $A(x)$ depends on the hidden state, it is not observable, and hence cannot be directly estimated. But Hsu et al. (2009) showed that under certain conditions there exists a

fully observable representation of the observable operator model. We now present a novel, fully reduced dimensional version of the observable representation.

2.1.2 The reduced dimension model

Define a random variable $y_t = U^\top x_t$, where U has orthonormal columns and is a matrix mapping from observations to the reduced dimension space.

We show below that

$$Pr(x_1, x_2, \ldots, x_t) = c_\infty^\top C(y_t) C_(y_{t-1}) \cdots C_(y_1) c_1 \tag{2.2}$$

holds where

$$\begin{aligned} c_1 &= \mu \\ c_\infty^\top &= \mu^\top \Sigma^{-1} \\ C(y) &= K(y)\Sigma^{-1} \end{aligned}$$

and $\mu = E(y_1)$, $\Sigma = E(y_2 \otimes y_1)$, and $K(a) = E(y_3 \otimes y_1 \otimes y_2)a$ are easy to estimate using the method of moments.[1]

The matrix U can be derived in several ways; Hsu et al. (2009) show that taking it to consist of the left singular vectors of P_{21} corresponding to the largest singular values gives good properties, where P_{21} is a matrix such that $[P_{21}]_{ij} = Pr[x_2 = i, x_1 = j]$. The matrix U and its properties will be discussed in more detail below.

Note that the model $(c_1, c_\infty, C(y))$ will be estimated using only trigrams. Once a model has been learned, the probability of any observed sequence $(x_1, x_2, ...x_t)$ can be computed using equation 2.2, or the conditional probability $Pr(x_t|x_1, x_2...x_{t-1})$ of the next observation x_t in a sequence can be computed by $Pr(x_t|x_{1:t-1}) = c_\infty^\top C(y_t) c_t$ with recursive updates $c_{t+1} = C(y_t)c_t / (c_\infty^\top C(y_t)c_t)$. The key term in the model is thus $C(y)$, which can be viewed as a tensor which takes as input the current observation x_t and produces a matrix which maps (after normalization) from the current "hidden state estimate" c_t to the next one c_{t+1}. More precisely, $c_{t+1} = (U^\top M)\widehat{h}_{t+1}(x_{1:t})$ is a linear function of the conditional expectation of the unobservable hidden state $\widehat{h}_{t+1}(x_{1:t})$, which is the conditional probability vector over states at time $t+1$.

2.1.3 Comparison to Hsu et al.

Hsu et al. (2009) derive a similar model which we state here for comparison.

$$Pr(x_1, x_2, ..., x_t) = b_\infty^\top B(x_t) B(x_{t-1}) \ldots B(x_1) b_1 \tag{2.3}$$

[1]Note that $K()$ is a tensor. When multiplied by a vector a, it produces a matrix. $K()$ is linear in each of the three reduced dimension observations, y_1, y_2 and y_3.

where

$$\begin{aligned} b_1 &= U^\top P_1 \\ b_\infty^\top &= P_1^\top (U^\top P_{21})^+ \\ B_x &= (U^\top P_{3x1})(U^\top P_{21})^+ \end{aligned}$$

and $[P_1]_i = Pr[x_1 = i]$, P_{21} as defined above, and $[P_{3x1}]_{ij} = Pr[x_3 = i, x_2 = x, x_1 = j]$ are the frequencies of unigrams, bigrams, and trigrams in the observed data. Note that the subscripts on x refer to their positions in trigrams of observations of the form (x_1, x_2, x_3).

Our major modeling change will be to replace $B(x)$ in equation 3.1 with the lower dimensional tensor $C(y)$ which depends on the reduced dimension projection $y \equiv U^\top x$ instead of the unreduced x. The models are easily related by the following lemma:

Lemma 1. *Assume the hidden state is of dimension k and the rank of M is also k. Then:*

$$\begin{aligned} Pr(x_1, x_2, \ldots, x_t) &= 1^\top A(x_t) A(x_{t-1}) \cdots A(x_1) \pi && (2.4) \\ &= b_\infty^\top B(x_t) B(x_{t-1}) \cdots B(x_1) b_1 && (2.5) \\ &= c_\infty^\top C(y_t) C(y_{t-1}) \cdots C(y_1) c_1 && (2.6) \end{aligned}$$

Where (2.5) requires $U^\top M$ to be invertible, and (2.6) requires range$(M) \subset$ range(U).[2]

Proof: Paper Jaeger (2000) showed (2.4), Hsu et al. (2009) showed (2.5), and (2.6) follows from a telescoping product of the following items:

$$\begin{aligned} c_1 &= U^\top M \, \pi \\ c_\infty^\top &= \mathbf{1}^\top \, (U^\top M)^{-1} \\ C(y) &= U^\top M \, A(x) \, (U^\top M)^{-1} \end{aligned}$$

where $y = U^\top x$. More details are given in Appendix A.2. □

Our expression (2.6) improves on that of Hsu et al. (2009) in three ways:

(a) By reducing the size of the matrix that is estimated, we can achieve a lower sample complexity. In particular, our sample complexity does not depend on the size of the vocabulary nor on the frequency distribution of the vocabulary.

(b) Since the conditions given in Hsu et al. (2009) are in terms of the transition matrix T, they can not be checked. We instead focus on conditions that are checkable from the data.

(c) Instead of using either a L1 error or a relative entropy error, we estimate the probabilities with relative accuracy. In other words, we show that $|\widehat{p} - p|/p$

[2] If the matrix U is formed from the left singular vectors of P_{21} corresponding to nonzero singular values, then it will satisfy this condition; See Hsu et al. (2009) lemma 2.

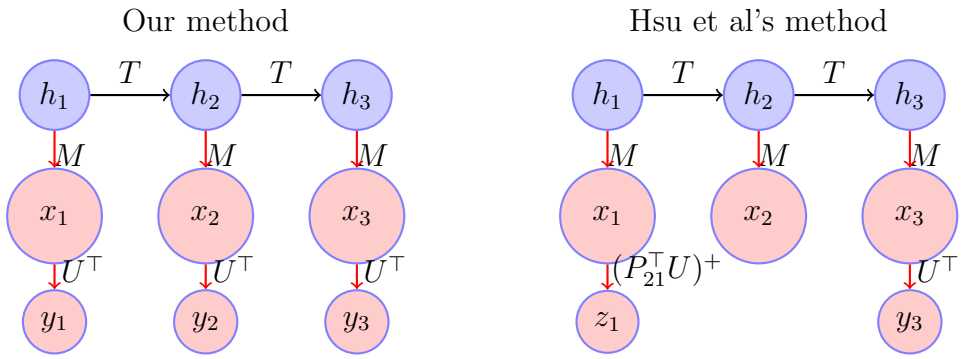

Figure 2.1: Two HMMs with states h_1, h_2, and h_3 which emit observations x_1, x_2, and x_3. On the left, they are further projected onto lower dimensional space with observations y_1, y_2, y_3 by U from which our core statistic C_y is computed based on $K = E(y_3 \otimes y_1 \otimes y_2)$ which is a $(k \times k \times k)$ tensor. On the right, x_1 is hit by $(P_{21}^\top U)^+$ to make a lower dimensional z_1, x_2 is left unchanged and x_3 has its dimension reduced by $U^\top$. These terminal leaves are then used by Hsu et al. (2009) to estimate their $B(x)$ via estimating $E(y_3 z_1^\top \delta_{x_2}^\top)$ which is a tensor of size $(k \times k \times v)$.

is smaller than ϵ. This often is a more useful bound than knowing $|\widehat{p} - p|$ is small. For example, it implies that computing conditional probabilities are off by less than 2ϵ. Both L1 and relative entropy errors can be computed from these bounds.

Our main theorem is weaker (as stated) than Hsu et al. (2009) in that we assume knowledge of U rather than estimating it from a thin SVD of P_{21} as they do. Since the accuracy lost when estimating U is identical to that given in their paper, we will not discuss it here.

2.2 Theorems

The remainder of this chapter presents one main theorem giving finite sample bounds for our reduced dimensional HMM estimation method. We first derive these in terms of properties of the first three moments of the reduced rank Y's, where Y is the random variable which takes on values of the reduced rank observation $y = U^\top x$. We then convert those bounds to be in terms of the estimates, rather than the unobservable true values, of the model.

Our general strategy of estimating $\Pr(x_t, x_{t-1}, \ldots, x_1)$ is via the method of moments. We have $\Pr()$ written in terms of $c_\infty^\top$, c_1 and $C(y_t)$. Since each of these three items can be written in terms of moments of the Y's we can plug in these moments

to generate an estimate of Pr(). Thus we can define:

$$\widehat{\Pr}(x_t, x_{t-1}, \ldots, x_1) = \widehat{c}_\infty^\top \widehat{C}(y_t)\widehat{C}(y_{t-1}) \cdots \widehat{C}(y_1)\widehat{c}_1 \tag{2.7}$$

where

$$\begin{aligned} \widehat{c}_1 &= \widehat{\mu} \\ \widehat{c}_\infty^\top &= \widehat{\mu}^\top \widehat{\Sigma}^{-1} \\ \widehat{C}(y) &= \widehat{K}(y)\widehat{\Sigma}^{-1} \end{aligned}$$

where $\widehat{\mu}$, $\widehat{\Sigma}$ and $\widehat{K}()$ are the empirical estimates of the first, second and third moments of the Y's, namely $\widehat{\mu} = \frac{1}{N}\sum_{i=1}^N Y_1^{(i)}$, $\widehat{\Sigma} = \frac{1}{N}\sum_{i=1}^N Y_1^{(i)} Y_2^{(i)\top}$, $\widehat{K}(y) = \frac{1}{N}\sum_{i=1}^N Y_1^{(i)} Y_3^{(i)\top} Y_2^{(i)\top} y$, where $Y^{(i)}$ indexes the N different independent observations of our data.

Definition 1. *Define Λ as the smallest element of μ, Σ^{-1} and $K()$. In other words,*

$$\Lambda \equiv \min\{\min_i |\mu_i|, \min_{i,j} |\Sigma_{ij}^{-1}|, \min_{i,j,l} |K_{ijl}|\}$$

where $K_{ijl} = K(\delta_j)_{il}$ are the elements of the tensor $K()$. Likewise we define the empirical version as

$$\widehat{\Lambda} \equiv \min\{\min_i |\widehat{\mu}_i|, \min_{i,j} |\widehat{\Sigma}_{ij}^{-1}|, \min_{i,j,l} |\widehat{K}_{ijl}|\}$$

Definition 2. *Define σ_k as the smallest singular value of Σ, and $\widehat{\sigma}_k$ the smallest singular value of $\widehat{\Sigma}$.*

The parameters Λ and σ_k will be central to our analysis. Theorem 1 gives sample complexity bounds on relative error in estimating the probability of a sequence being generated from an HMM as a function of Λ and σ_k, and the following lemmas reformulate those bounds into a more useful form in terms of their estimates. As quantified and proved below, both Λ and σ_k must be "sufficiently large"; when they approach zero one loses the ability to accurately estimate the model.

If $\sigma_k = 0$ then $U^\top M$ will not be invertible, and one cannot infer the full information content of the hidden state from its associated observation, violating the condition required in Hsu et al. (2009) for (2.5) to hold. As σ_k becomes increasingly close to zero, it becomes increasingly hard to identify the hidden state, and more observations are required. Problems with small σ_k are intrinsically difficult. As has been pointed out by Hsu et al. (2009), some problems of estimating HMM's are equivalent to the parity problem Terwijn (2002a). For such data, our algorithm need not perform well. For parity-like problems, σ_k is in fact zero, or close to it; hence we end up with a useless bound for such hard problems.

If Λ is close to zero, then even if the absolute error is small, the relative error can be arbitrarily large, as it involves dividing by the small true value of the parameter being estimated. Fortunately, as discussed below, since Λ depends on the somewhat arbitrary matrix U, one can shift Λ away from zero by rotating and rescaling U.

The proof of Theorem 1 is based on the idea that if we can estimate each term in μ, Σ and $K()$ accurately on an absolute scale (which will follow from basic central limit like theorems) then we can estimate them on a relative scale if Λ is large. Hence, our main condition is that Λ is bounded away from zero. In fact, if we take the usual statistical limit of having the sample size N go to infinity and holding everything else constant, then:

$$\left| \frac{\widehat{\Pr}(x_1, \ldots, x_t)}{\Pr(x_1, \ldots, x_t)} - 1 \right| \leq \frac{18kt}{\sigma_k^2 \Lambda \sqrt{N}} \sqrt{\log(k/\delta)}$$

with probability greater than $1 - \delta$.

The following theorem gives the finite sample bound in terms of a sample complexity:

Theorem 1. *Let X_t be generated by an $k \geq 2$ state HMM. Suppose we are given a U which has the property that range$(M) \subset$ range(U) and $|U_{ij}| \leq 1$. Suppose we use equation (2.7) to estimate the probability based on N independent triples. Then*

$$N \geq \frac{128k^2}{(\sqrt[2t+3]{1+\epsilon} - 1)^2 \, \Lambda^2 \sigma_k^4} \log\left(\frac{2k}{\delta}\right) \tag{2.8}$$

implies that

$$1 - \epsilon \leq \left| \frac{\widehat{\Pr}(x_1, \ldots, x_t)}{\Pr(x_1, \ldots, x_t)} \right| \leq 1 + \epsilon$$

holds with probability at least $1 - \delta$.

Before proceeding with the proof of this theorem, we present and prove two corollaries that correspond directly to Theorems 6 and 7 of Hsu et al. (2009).

Corollary 1. *Assume Theorem 1 holds, then with probability at least $1 - \delta$,*

$$\sum_{x_1, \ldots, x_t} |\widehat{\Pr}(x_t, \ldots, x_t) - \Pr(x_1, \ldots, x_t)| \leq \epsilon$$

Proof of Corollary 1: We have

$$\begin{aligned}
& 1 - \epsilon \leq \left| \frac{\widehat{\Pr}(x_1, \ldots, x_t)}{\Pr(x_1, \ldots, x_t)} \right| \leq 1 + \epsilon \\
\Rightarrow & \left| \frac{\widehat{\Pr}(x_1, \ldots, x_t)}{\Pr(x_1, \ldots, x_t)} - 1 \right| \leq \epsilon \\
\Rightarrow & \left| \widehat{\Pr}(x_1, \ldots, x_t) - \Pr(x_1, \ldots, x_t) \right| \leq \epsilon \Pr(x_1, \ldots, x_t) \\
\Rightarrow & \sum_{x_1, \ldots, x_t} \left| \widehat{\Pr}(x_1, \ldots, x_t) - \Pr(x_1, \ldots, x_t) \right|
\end{aligned}$$

$$\leq \epsilon \sum_{x_1,\ldots,x_t} \Pr(x_1,\ldots,x_t)$$
$$\Rightarrow \sum_{x_1,\ldots,x_t} \left|\widehat{\Pr}(x_1,\ldots,x_t) - \Pr(x_1,\ldots,x_t)\right| \leq \epsilon$$

□

For the next corollary, let $KL(A||B)$ be the Kullback-Leibler divergence between probability distributions A and B, so $KL(A||B) = \sum_x \ln\left(\frac{A(x)}{B(x)}\right) A(x)$.

Corollary 2. *Assume Theorem 1 holds, then we have*

$$KL(\Pr(x_t|x_1,\ldots x_{t-1})||\widehat{\Pr}(x_t|x_1,\ldots x_{t-1})) = E\left(\ln \frac{\Pr(x_t|x_1,\ldots x_{t-1})}{\widehat{\Pr}(x_t|x_1,\ldots x_{t-1})}\right) \leq 6\epsilon$$

Proof of Corollary 2: We have

$$1-\epsilon \leq \left|\frac{\widehat{\Pr}(x_1,\ldots,x_t)}{\Pr(x_1,\ldots,x_t)}\right| \leq 1+\epsilon$$
$$\Rightarrow 1-\epsilon \leq \left|\frac{\widehat{\Pr}(x_t|x_{1:t-1})\widehat{\Pr}(x_{1:t-1})}{\Pr(x_t|x_{1:t-1})\Pr(x_{1:t-1})}\right| \leq 1+\epsilon$$
$$\Rightarrow \frac{1-\epsilon}{1+\epsilon} \leq \left|\frac{\widehat{\Pr}(x_t|x_{1:t-1})}{\Pr(x_t|x_{1:t-1})}\right| \leq \frac{1+\epsilon}{1-\epsilon}$$

and using the fact that for small enough x we have $\frac{1+x}{1-x} \leq 1+3x$ and $1-3x \leq \frac{1-x}{1+x}$, plus the fact that $\epsilon_0 \leq \frac{\epsilon}{6}$ we have

$$\Rightarrow 1-3\epsilon \leq \left|\frac{\widehat{\Pr}(x_t|x_{1:t-1})}{\Pr(x_t|x_{1:t-1})}\right| \leq 1+3\epsilon$$
$$\Rightarrow \frac{1}{1+3\epsilon} \leq \left|\frac{\Pr(x_t|x_{1:t-1})}{\widehat{\Pr}(x_t|x_{1:t-1})}\right| \leq \frac{1}{1-3\epsilon}$$

and using a similar fact from above that for small enough x, $\frac{1}{1-x} \leq 1+2x$, we get

$$\Rightarrow \left|\frac{\Pr(x_t|x_{1:t-1})}{\widehat{\Pr}(x_t|x_{1:t-1})}\right| \leq 1+6\epsilon$$
$$\Rightarrow \ln\left[\frac{\widehat{\Pr}(x_t|x_{1:t-1})}{\Pr(x_t|x_{1:t-1})}\right] \leq \ln(1+6\epsilon) \leq 6\epsilon$$

$$\Rightarrow \sum_{x_1,\ldots,x_t} \Pr(x_1,\ldots,x_t) \ln \left[\frac{\widehat{\Pr}(x_t|x_{1:t-1})}{\Pr(x_t|x_{1:t-1})}\right]$$
$$\leq 6\epsilon \sum_{x_1,\ldots,x_t} \Pr(x_1,\ldots,x_t)$$
$$\Rightarrow E \ln \left[\frac{\widehat{\Pr}(x_t|x_{1:t-1})}{\Pr(x_t|x_{1:t-1})}\right] \leq 6\epsilon$$

□

Define $J \equiv 2k\sqrt{\frac{2\log\frac{2k}{\delta}}{N}}$ to simplify the following statements. The proof proceeds in two steps. First lemma 2 converts the sample complexity bound into a more useful bounds on Λ and σ_k. Then lemma 3 uses these bounds to show the theorem.

Lemma 2. *If*

$$N \geq \frac{128k^2}{(\sqrt[2t+3]{1+\epsilon}-1)^2\Lambda^2\sigma_k^4}\log\left(\frac{2k}{\delta}\right)$$

then

$$\Lambda \geq \frac{3J}{\sigma_k^2(\sqrt[2t+3]{1+\epsilon}-1)} \tag{2.9}$$
$$\sigma_k \geq 4J \tag{2.10}$$

The proof is straightforward and given in the appendix.

Lemma 3. *If equation (2.8) of Theorem 1 is replaced by (2.9) and (2.10) then the results of the theorem follow.*

Proof of Lemma 3: Our estimator (see equation 2.7) can be written as

$$\widehat{\Pr}(x_1,\ldots,x_t) = \widehat{\mu}^\top \widehat{\Sigma}^{-1}\widehat{K}(y_t)\widehat{\Sigma}^{-1}\cdots\widehat{K}(y_1)\widehat{\Sigma}^{-1}\widehat{\mu}$$

We can rewrite this matrix product as

$$\widehat{\Pr}(x_1,\ldots,x_t) =$$
$$\sum_{i_1=1}^{k}\cdots\sum_{i_{2t+3}=1}^{k} [\widehat{\mu}]_{i_1}[\widehat{\Sigma}^{-1}]_{i_1,i_2}[\widehat{K}(y_t)]_{i_2,i_3}[\widehat{\Sigma}^{-1}]_{i_3,i_4}$$
$$\cdots[\widehat{\mu}]_{i_{2t+3}}$$

The components $[\widehat{K}(y)]_{a,b}$ can be written as a scalar sum as:

$$[\widehat{K}(y)]_{a,b} = y_1[\widehat{K}]_{a,b,1} + y_2[\widehat{K}]_{a,b,2} + \ldots + y_k[\widehat{K}]_{a,b,k}$$

So,

$$\widehat{\Pr}(x_1,\ldots,x_t) =$$

$$\sum_{\substack{i_1,\ldots,i_{2t+3} \\ j_1,\ldots,j_t}} [\widehat{\mu}]_{i_1} [\widehat{\Sigma}^{-1}]_{i_1,i_2} [\widehat{K}]_{i_2,i_3,j_1} [y_t]_{j_1} \cdot [\widehat{\Sigma}^{-1}]_{i_3,i_4} [\widehat{K}]_{i_5,i_6,j_2} [y_{t-1}]_{j_2} \cdots [\widehat{\mu}]_{i_{2t+3}} \tag{2.11}$$

This is a sum of a product of scalars. Lemma 8 (stated precisely and proven in the appendix) shows that accuracy of our estimates of all elements of μ, Σ^{-1} and $K()$ are bounded by $3J/\sigma_k^2$ with probability $1-\delta$.

Each term in 2.11 can be rewritten as

$$\widehat{\theta} \;=\; \theta\left(1+\frac{\widehat{\theta}-\theta}{\theta}\right)$$

and so our products can be thought of as, instead of a product of observed quantities, the product of the theoretical quantities times some relative error term. We can bound this relative error term for all entries, which will allow it to factor out nicely over all summands, giving us a relative error term for our overall probability.

Again thinking of θ as a generic item in μ, Σ, or $K()$, then above has shown that $|\widehat{\theta}-\theta| \leq 3J/\sigma_k^2$ and so the relative error of each term is bounded as

$$1-\frac{3J}{\sigma_k^2\theta} \leq \frac{\widehat{\theta}}{\theta} \leq 1+\frac{3J}{\sigma_k^2\theta}$$

which will hold for all terms with probability $1-\delta$. Since $|\theta| \geq \Lambda$, we see that

$$1-\frac{3J}{\sigma_k^2\Lambda} \leq \frac{\widehat{\theta}}{\theta} \leq 1+\frac{3J}{\sigma_k^2\Lambda}$$

Since our $\widehat{\Pr}()$ is a product of $2t+3$ such terms, we see that

$$\left(1-\frac{3J}{\sigma_k^2\Lambda}\right)^{2t+3} \leq \frac{\widehat{\Pr}()}{\Pr()} \leq \left(1+\frac{3J}{\sigma_k^2\Lambda}\right)^{2t+3}$$

So by our bound on Λ, we have

$$1-\epsilon \leq \frac{\widehat{\Pr}()}{\Pr()} \leq 1+\epsilon$$

holds with probability $1-\delta$. □

The sample complexity bound in Theorem 1 relies on knowing unobserved parameters of the problem. To avoid this, we modify Lemma 3 to make it observable. In other words, we convert the assumptions of sample complexity into a checkable condition.

Corollary 3. *Let X_t be generated by an $k \geq 2$ state HMM. Suppose we are given a U which has the property that range$(M) \subset$ range(U). Suppose we use equation (2.7) to estimate the probability based on N independent triples. Then with probability $1-\delta$,*

if the following two inequalities hold

$$\widehat{\Lambda}\,\widehat{\sigma}_k^2 \geq \left(12k + \frac{6k}{(\sqrt[2t+3]{1+\epsilon}-1)}\right)\sqrt{\frac{2\log\frac{2k}{\delta}}{N}} \tag{2.12}$$

$$\widehat{\sigma}_k \;\geq\; 10k\sqrt{\frac{2\log\frac{2k}{\delta}}{N}}. \tag{2.13}$$

then

$$1-\epsilon \leq \left|\frac{\widehat{\Pr}(x_1,\ldots,x_t)}{\Pr(x_1,\ldots,x_t)}\right| \leq 1+\epsilon$$

Proof:

Two technical lemmas are needed for this corollary: Lemma 8 and Lemma 9. They are stated and proved in appendix A.2. Lemma 8 basically says that with high probability, each element of μ, Σ and $K()$ is estimated accurately. This is then used in Lemma 9 to show that Λ and σ_k are estimated accurately.

Recall that the theorems in this chapter are statements of high probability, where a bound holds with probability $1-\delta$. Define the event $\mathcal{A}$ to be the set where all the estimates given in Lemma 8 hold. On this event from Lemma 9 we know $\sigma_k \geq \frac{4}{5}\widehat{\sigma}_k$, so $\sigma_k^2 \geq \frac{1}{2}\widehat{\sigma}_k^2$. Hence

$$\widehat{\Lambda} \geq \frac{6k}{\sigma_k^2(\sqrt[2t+3]{1+\epsilon}-1)}\sqrt{\frac{2\log\frac{2k}{\delta}}{N}} + \frac{6k}{\sigma_k^2}\sqrt{\frac{2\log\frac{2k}{\delta}}{N}},$$

thus on the set $\mathcal{A}$ if (2.12) and (2.13) hold, then we see that (2.9) and (2.10) both hold and so we can apply Theorem 1. We can now use Theorem 1 to generate our claim on the accuracy of our probability bound. Technically, this proof as given only shows that our corollary holds with probability $1-2\delta$. But since the set where Theorem 1 fails is exactly $\mathcal{A}^c$, the probability lower bound is $1-\delta$.

□

The advantage of the corollary is that the left hand sides of the two conditions are observable and the right hand sides involve known quantities. Hence one can tell if the condition is true or not–it doesn't require knowing unobserved parameters.

2.3 Discussion: effect of Λ and σ_k on accuracy

As discussed above, σ_k and Λ have different effects on sample complexity. As σ_k approaches zero, model estimation becomes intrinsically hard; some problems do not admit easy estimation. In contrast, role of Λ in sample complexity is more of an artifact. As Λ approaches zero, the relative error can be arbitrarily large, even if

the estimated model is good in the sense that the probability estimates are highly accurate.

The problem with Λ can be addressed in different ways. In this section, we show that estimating a likelihood ratio rather than the sequence probabilities improves relative accuracy bounds. An alternate approach, which we do not pursue here, relies on the observation that Λ depends on the (underspecified) matrix $\hat{U}$, and that one can thus search for a rotation and rescaling of the matrix $\hat{U}$ that increases Λ.

2.3.1 Likelihood instead of probabilities

Obscure words correspond to rows of the observation matrix with very small values throughout the row. If we were interested in only estimating the probability of such a word, then these are the easy words–basically guess zero or close to it. But, since we would like to estimate the relative probability accurately, these words are the most challenging. Further, such small probabilities would make computing conditional probabilities unstable since they would then become basically "0/0." Further, since the values are all small in M and in U, they do not significantly improve our estimates of μ, Σ and $K()$ since they are essentially zeros. Both of these problems can be fixed by considering the problem of estimating a likelihood ratio instead of a probability. So define:

$$\lambda_q(x_1, \ldots, x_t) = \frac{Pr(x_1, x_2, \ldots, x_t)}{P_1(x_1)P_1(x_2)\cdots P_1(x_t)}$$

The $P_1(x)$ could be taken to be the marginal probability of observing x. It does not, in fact, have to be a probability–just any weighting which helps condition our matrix Σ and our tensor $K()$. We can then use a modified version of M and U in all our existing lemma's and theorems. The precise statement of these modified versions are in the appendix. What changes is that now Λ is much larger and hence our relative accuracy will be greatly improved. This fact is shown in the empirical section.

2.3.2 Empirical estimates of Λ and σ_k

Figure 2.2 shows estimates of $\widehat{\Lambda}$ and $\widehat{\sigma}_k$, using the Internet as the corpus as summarized in the Google n-gram dataset[3], which contains frequencies of the most common 1-grams to 5-grams occurring on the web. Details on how the figures were generated can be found in Appendix A.2. As the size, k, of the reduced dimension space is increased, smaller and smaller singular values, σ_k, occur in the model, and the value Λ of the smallest parameter in the model decreases. Empirically, both fall off with a power of k, giving straight lines on the log-log plot. This data indicates a large sample complexity, the reduction of which will be a focus of future work.

[3]http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html

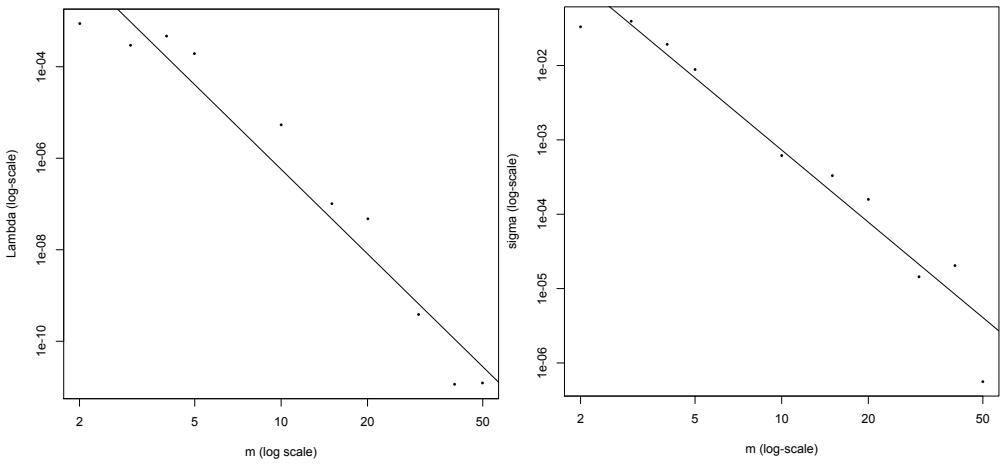

Figure 2.2: **First graph:** Λ vs k, generated using vocabulary size $20,000$, Slope ≈ -6. **Second graph:** $\widehat{\sigma}$ vs k, generated using vocabulary size of $10,000$, Slope ≈ -3.2

2.4 Prior work and conclusion

Recently, ideas have been proposed that push spectral learning of HMMs in several different directions. Boots et al. (2010) provides a kernelized spectral algorithm that allows for learning an HMM in any domain in which there exists a kernel. This allows for learning of an HMM with continuous output without the need for discretization. Boots and Gordon (2011) provides an analogous algorithm that enables online learning for Transformed Predictive State Representations, and hence the setup in Hsu et al. (2009). Finally, Siddiqi et al. (2010) directly extends Hsu et al. (2009) by relaxing the requirement that the transition matrix T be of rank k, but instead allows rank less than k, creating a Reduced-Rank HMM (RR-HMM), and then applying the algorithm from Hsu et al. (2009) to learn the observable representation of this RR-HMM.

All of the above extensions preserve the basic structure of the tensor $B(x)$, which updates the hidden state estimate (or more precisely, a linear transformation of it) based on the most recent observation x. In this paper, we replace $B(x)$ with a tensor $C(y)$, which updates the hidden state estimate using a low dimensional projection y of the observation x. $C(y)$ contains only k^3 terms, in contrast to the $k^2 v$ terms contained in $B(x)$. Reducing the number of parameters to be estimated has both computational and statistical efficiency advantages, but requires some changes to the proofs in Hsu et al. (2009). While making these changes, we also give proofs that are simpler, that only use conditions that are checkable from the data, and that bound the relative, rather than absolute error.

This paper focused on the simplest case, in which HMMs have discrete states and

discrete observations and in which the observations are reduced to the same sized space as the hidden state, but our approach can be generalized in all of the ways described above.

We have presented an improved spectral method for estimating HMMs. By using a tensor $C(y)$ that depends on the reduced rank y instead of the full observed x in the $B(x)$ tensor used by Hsu et al. (2009), we reduced the number of parameters to be estimated by a factor of the ratio of the size of the vocabulary divided by the size of the hidden state. This reduction has corresponding benefits in the sample complexity. We also showed that the sample complexity depends critically upon σ_k, the smallest singular value of the covariance matrix Σ. As σ_k becomes small, the HMM becomes increasingly hard to identify, and increasing numbers of samples are needed.

3

Application: Spectral learning of HMMs for Trees

Recently there has been substantial interest in using spectral methods to learn generative sequence models like HMMs. Spectral methods are attractive as they provide globally consistent estimates of the model parameters and are very fast and scalable, unlike EM methods, which can get stuck in local minima. In this paper, we present a novel extension of this class of spectral methods to learn dependency tree structures. We propose a simple yet powerful latent variable generative model for dependency parsing, and a spectral learning method to efficiently estimate it. As pilot experimental evaluations, 1) we use the spectral tree probabilities estimated by our model to re-rank the outputs of a discriminative parser and 2) use spectral estimates of probabilities of edges and sub-trees as features in a discriminative parser. Our approach reduces the error of the baseline parser by up to 7.3%.

3.1 Introduction

Markov models have been for two decades a workhorse of statistical pattern recognition with applications ranging from speech to vision to language. Adding latent variables to these models gives us additional modeling power and have shown success in applications like POS tagging Merialdo (1994), speech recognition Rabiner (1989) and object recognition Quattoni et al. (2004). However, this comes at the cost that the resulting parameter estimation problem becomes non-convex and techniques like EM Dempster et al. (1977) which are used to estimate the parameters can only lead to locally optimal solutions.

Recent work by Hsu et al. (2009) has shown that globally consistent estimates of the parameters of the HMMs can be found by using spectral methods, particularly by singular value decomposition (SVD) of appropriately defined linear systems. They avoid the NP Hard problem of the global optimization problem of the HMM parameters Terwijn (2002b), by putting restrictions on the smallest singular value of

Work from this chapter appears in Rodu et al. (2012)

the HMM parameters. The main intuition behind the model is that, although the observed data (i.e. words) seems to live in a very high dimensional space but in reality they live in a very low dimensional space (size $k \sim 30 - 50$) and an appropriate eigen decomposition of the observed data will reveal the underlying low dimensional dynamics and thereby revealing the parameters of the model. Besides ducking the NP hard problem, the spectral methods are very fast and scalable to train compared to EM methods.

In this paper we generalize the approach of Hsu et al. (2009) to learn dependency tree structures with latent variables.[1] Petrov et al. (2006); Musillo and Merlo (2008) have shown that learning PCFGs and dependency grammars respectively with latent variables can produce parsers with very good generalization performance. However, both these approaches rely on EM for parameter estimation and can benefit from using spectral methods.

We propose a simple yet powerful latent variable generative model for use with dependency parsing which has one hidden node for each word in the sentence, like the one shown in figure 3.1 and work out the details for the parameter estimation of the corresponding spectral learning model. At a very high level, the parameter estimation of our model involves collecting unigram, bigram and trigram counts sensitive to the underlying dependency structure of the given sentence.

Recently Luque et al. (2012) have also proposed a spectral method for dependency parsing, however they deal with *horizontal Markovization* and use hidden states to model sequential dependencies within a word's sequence of children. In contrast with that, in this paper, we propose a spectral learning algorithm where latent states are not restricted to HMM-like distributions of modifier sequences for a particular head, but instead allow information to be propagated through the entire tree.

More recently Cohen et al. (2012) have proposed a spectral method for learning PCFGs.

Its worth noting that recent work by Parikh et al. (2011) also extends Hsu et al. (2009) to latent variable dependency trees like us but under the restrictive conditions that model parameters are trained for a specified, albeit arbitrary, tree topology.[2] In other words, all training sentences and test sentences must have identical tree topologies. By doing this they allow for node-specific model parameters, but must retrain the model entirely when a different tree topology is encountered. Our model on the other hand allows the flexibility and efficiency of processing sentences with a variety of tree topologies from a single training run.

Most of the current state-of-the-art dependency parsers are discriminative parsers Koo et al. (2008); McDonald (2006) due to the flexibility of representations which can be used as features leading to better accuracies and the ease of reproducibility of results. However, unlike discriminative models, generative models can exploit unlabeled data. Also, as is common in statistical parsing, re-ranking the outputs of a parser leads to

[1] Actually, instead of using the model by Hsu et al. (2009) we work with a related model proposed in chapter 2 which addresses some of the shortcomings of the earlier model which we detail below.

[2] This can be useful in modeling phylogeny trees for instance, but precludes most NLP applications, since there is a need to model the full set of different tree topologies possible in parsing.

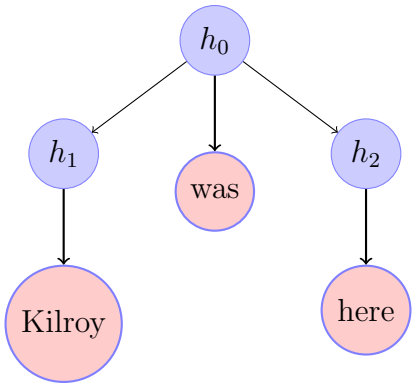

Figure 3.1: Sample dependency parsing tree for "Kilroy was here"

significant reductions in error Collins and Koo (2005).

Since our spectral learning algorithm uses a generative model of words given a tree structure, it can score a tree structure i.e. its probability of generation. Thus, it can be used to re-rank the n-best outputs outputs of a given parser; note that unlike the standard discriminative re-rankers, our approach is generative.[3] In addition to that, we can use our edge factored spectral probability scores as features in a discriminative parser.

The remainder of the paper is organized as follows. In the next section we introduce the notation and give a brief overview of the spectral algorithm for learning HMMs (see for instance Hsu et al. (2009) and chapter 2). In Section 3 we describe our proposed model for dependency parsing in detail and work out the theory behind it. Section 4 provides experimental evaluation of our model on Penn Treebank data. We demonstrate the utility of our model by re-ranking the n-best outputs of a standard discriminative (the MST parser) McDonald (2006) and by using edge spectral probabilities as features in a discriminative parser. We conclude with a brief summary and future avenues for research.

3.2 Spectral algorithm for learning HMMs

In this section we describe the spectral algorithm for learning HMMs.[4]

[3]Sangati et al. (2009) also propose a carefully defined generative re-ranker for Italian dependency parsing which outperforms discriminative re-ranker.

[4]As mentioned earlier, we use the model in chapter 2 which is conceptually similar to the one by Hsu et al. (2009), but does further dimensionality reduction and thus has lower sample complexity. Also, critically, the fully reduced dimension model that we use generalizes much more cleanly to trees.

3.2.1 Notation

The HMM that we consider in this section is a sequence of hidden states $h \in \{1, \ldots, k\}$ that follow the Markov property:

$$\Pr(h_t|h_1, \ldots, h_{t-1}) = \Pr(h_t|h_{t-1})$$

and a sequence of observations $x \in \{1, \ldots, n\}$ such that

$$\Pr(x_t|x_1, \ldots, x_{t-1}, h_1, \ldots, h_t) = \Pr(x_t|h_t)$$

The parameters of this HMM are:

- A vector π of length k where $\pi_i = \Pr(h_1 = i)$: The probability of the start state in the sequence being i.
- A matrix T of size $k \times k$ where $T_{i,j} = \Pr(h_{t+1} = i|h_t = j)$: The probability of transitioning to state i, given that the previous state was j.
- A matrix M of size $n \times k$ where $M_{i,j} = \Pr(x = i|h = j)$: The probability of state h emitting observation x.

Define δ_j to be the vector of length n with a 1 in the j^{th} entry and 0 everywhere else, and $\text{diag}(v)$ to be the matrix with the entries of v on the diagonal and 0 everywhere else.

The joint distribution of a sequence of observations $x_1, \ldots, x_m$ and a sequence of hidden states $h_1, \ldots, h_m$ is:

$$\begin{aligned}\Pr(x_1, \ldots, &x_m, h_1, \ldots, h_m) \\ &= \pi_{h_1} \prod_{j=2}^{m-1} T_{h_j, h_{j-1}} \prod_{j=1}^{m} M_{x_j, h_j}\end{aligned}$$

Now, we can write the marginal probability of a sequence of observations as

$$\begin{aligned}\Pr(x_1, \ldots &x_m) \\ &= \sum_{h_1, \ldots, h_m} \Pr(x_1, \ldots, x_m, h_1, \ldots, h_m)\end{aligned}$$

which can be expressed in matrix form[5] as:

$$\Pr(x_1, \ldots, x_m) = \mathbf{1}^\top A(x_m) A(x_{m-1}) \cdots A(m_1) \pi$$

where $A(x_m) \equiv T\text{diag}(M^\top x_m)$, and $\mathbf{1}$ is a k-dimensional vector with every entry equal to 1.

[5]This is essentially the matrix form of the standard dynamic program (forward algorithm) used to estimate HMMs.

$A(x_m)$ is called an "observation operator", and is effectively a third order tensor, giving the distribution vector over states at time $m+1$ as a function of the state distribution vector at the current time m and the current observation x_m. Since $A(x_m)$ depends on the hidden state, it is not observable, and hence cannot be directly estimated. However chapter 2 showed that under certain conditions there exists a fully observable representation of the observable operator model.

3.2.2 Fully Observable Representation

Before presenting the model, we need to address a few more points. First, let U be a "representation matrix" (eigenfeature dictionary) which maps each observation to a reduced dimension space ($n \to k$) that satisfies the conditions:

- $U^\top M$ is invertible
- $|U_{ij}| < 1$.

Hsu et al. (2009) and chapter 2 discuss U in more detail, but U can, for example, be obtained by the SVD of the bigram probability matrix (where $P_{ij} = \Pr(x_{t+1} = i | x_t = j)$) or by doing CCA on neighboring n-grams Dhillon et al. (2011).

Letting $y_i = U^\top x_i$, we have

$$\begin{aligned} &\Pr(x_1, \ldots, x_m) \\ &\quad = c_\infty^\top C(y_m) C(y_{m-1}) \ldots C(y_1) c_1 \end{aligned} \tag{3.1}$$

where

$$\begin{aligned} c_1 &= \mu \\ c_\infty &= \mu^\top \Sigma^{-1} \\ C(y) &= K(y) \Sigma^{-1} \end{aligned}$$

and μ, Σ and K, described in more detail below, are quantities estimated by frequencies of unigrams, bigrams, and trigrams in the observed (training) data.

Under the assumption that data is generated by an HMM, the distribution $\hat{p}$ obtained by substituting the estimated values $\hat{c}_1$, $\hat{c}_\infty$, and $\widehat{C}(y)$ into equation (3.1) converges to p sufficiently fast as the amount of training data increases, giving us consistent parameter estimates. For details of the convergence proof, please see Hsu et al. (2009) and chapter 2.

3.3 Spectral algorithm for Learning Dependency Trees

In this section, we first describe a simple latent variable generative model for dependency parsing. We then define some extra notation and finally present the details of the corresponding spectral learning algorithm for dependency parsing, and prove that our learning algorithm provides a consistent estimation of the marginal probabilities.

It is worth mentioning that an alternate way of approaching the spectral estimation of latent states for dependency parsing is by converting the dependency trees into linear sequences from root-to-leaf and doing a spectral estimation of latent states using Hsu et al. (2009). However, this approach would not give us the correct probability distribution over trees as the probability calculations for different paths through the trees are not independent. Thus, although one could calculate the probability of a path from the root to a leaf, one cannot generalize from this probability to say anything about the neighboring nodes or words. Put another way, when a parent has more than the one descendant, one has to be careful to take into account that the hidden variables at each child node are all conditioned on the hidden variable of the parent.

3.3.1 A latent variable generative model for Dependency Parsing

In the standard setting, we are given training examples where each training example consists of a sequence of words $x_1, \ldots, x_m$ together with a dependency structure over those words, and we want to estimate the probability of the observed structure. This marginal probability estimates can then be used to build an actual generative dependency parser, re-rank the outputs of another parser or to augment the feature set of a discriminative parser.

As in the conventional HMM described in the previous section, we can define a simple latent variable first order dependency parsing model by introducing a hidden variable h_i for each word x_i. The joint probability of a sequence of observed nodes $x_1, \ldots, x_m$ together with hidden nodes $h_1, \ldots, h_m$ can be written as

$$\begin{aligned} &\Pr(x_1, \ldots, x_m, h_1, \ldots, h_m) \\ &\qquad = \pi_{h_1} \prod_{j=2}^{m} t_{d(j)}(h_j | h_{pa(j)}) \prod_{j=1}^{m} o(x_j | h_j) \end{aligned} \tag{3.2}$$

where $pa(j)$ is the parent of node j and $d(j) \in \{L, R\}$ indicates whether h_j is a left or a right node of $h_{pa(j)}$. For simplicity, the number of hidden and observed nodes in our tree are the same, however they are not required to be so.

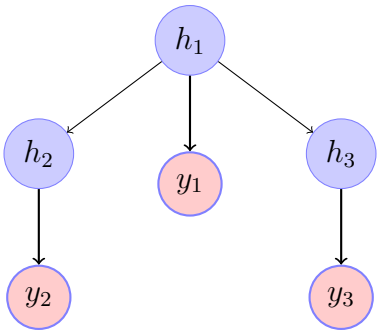

Figure 3.2: Dependency parsing tree with observed variables y_1, y_2, and y_3.

As is the case with the conventional HMM, the parameters used to calculate this joint probability are unobservable, but it turns out that under suitable conditions a fully observable model is also possible for the dependency tree case with the parameterization as described below.

3.3.2 Parameters

We will define both the theoretical representations of our observable parameters, and the sampling versions of these parameters. Note in all cases the estimated versions are unbiased estimates of the theoretical quantities.

Define T_d and T_d^u where $d \in \{L, R\}$ to be the hidden state transition matrices from parent to left or right child, and from left or right child to parent (hence the u for 'up'), respectively. In other words (referring to Figure 3.2)

$$\begin{aligned} T_R &= t(h_3|h_1) \\ T_L &= t(h_2|h_1) \\ T_R^u &= t(h_1|h_3) \\ T_L^u &= t(h_1|h_2) \end{aligned}$$

Let $U_{x(i)}$ be the i^{th} entry of vector $U^\top x$ and $G = U^\top M$. Further, recall the notation $\text{diag}(v)$, which is a matrix with elements of v on its diagonal, then:

- Define the k-dimensional vector μ (unigram counts):

$$\begin{aligned} \mu &= G\pi \\ [\hat{\mu}]_i &= \sum_{u=1}^{n} \bar{c}(u) U_{u(i)} \end{aligned}$$

  where $\bar{c}(u) = \frac{c(u)}{N_1}$, $c(u)$ is the count of observation u in the training sample, and $N_1 = \sum_{u \in n} c(u)$.

- Define the $k \times k$ matrices Σ_L and Σ_R (*left child-parent* and *right child-parent* bigram counts):

$$
\begin{aligned}
[\hat{\Sigma}_L]_{i,j} &= \sum_{u=1}^{n}\sum_{v=1}^{n} \bar{c}_L(u,v) U_{u(j)} U_{v(i)} \\
\Sigma_L &= G T_L^u \mathrm{diag}(\pi) G^\top \\
[\hat{\Sigma}_R]_{i,j} &= \sum_{u=1}^{n}\sum_{v=1}^{n} \bar{c}_R(u,v) U_{u(j)} U_{v(i)} \\
\Sigma_R &= G T_R^u \mathrm{diag}(\pi) G^\top
\end{aligned}
$$

  where $\bar{c}_L(u,v) = \frac{c_L(u,v)}{N_{2_L}}$, $c_L(u,v)$ is the count of bigram (u,v) where u is the left child and v is the parent in the training sample, and $N_{2_L} = \sum_{(u,v)\in n\times n} c_L(u,v)$. Define $\bar{c}_R(u,v)$ similarly.

- Define $k \times k \times k$ tensor K (*left child-parent-right child* trigram counts):

$$
\begin{aligned}
\hat{K}_{i,j,l} &= \sum_{u=1}^{n}\sum_{v=1}^{n}\sum_{w=1}^{n} \bar{c}(u,v,w) U_{w(i)} U_{u(j)} U_{v(l)} \\
K(y) &= G T_L \mathrm{diag}(G^\top y) T_R^u \mathrm{diag}(\pi) G^\top
\end{aligned}
$$

  where $\bar{c}(w,u,v) = \frac{c(w,u,v)}{N_3}$, $c(w,u,v)$ is the count of bigram (w,u,v) where w is the left child, u is the parent and v is the right child in the training sample, and $N_3 = \sum_{(w,u,v)\in n\times n\times n} c(w,u,v)$.

- Define $k \times k$ matrices Ω_L and Ω_R (skip-bigram counts (left child-right child) and (right child-left child)) [6]:

$$
\begin{aligned}
[\hat{\Omega}_L]_{i,j} &= \sum_{u=1}^{n}\sum_{v=1}^{n}\sum_{w=1}^{n} \bar{c}(u,v,w) U_{w(i)} U_{u(j)} \\
\Omega_L &= G T_L T_R^u \mathrm{diag}(\pi) G^\top \\
[\hat{\Omega}_R]_{i,j} &= \sum_{u=1}^{n}\sum_{v=1}^{n}\sum_{w=1}^{n} \bar{c}(u,v,w) U_{w(j)} U_{u(i)} \\
\Omega_R &= G T_R T_L^u \mathrm{diag}(\pi) G^\top
\end{aligned}
$$

3.3.3 Model estimation algorithm

Using the above definitions, we can estimate the parameters of the model, namely $\mu, \Sigma_L, \Sigma_R, \Omega_L, \Omega_R$ and K, from the training data and define observables useful for the

[6] Note than $\Omega_R = \Omega_L^T$, which is not immediately obvious from the matrix representations.

dependency model as[7]

$$
\begin{aligned}
c_1 &= \mu \\
c_\infty^T &= \mu^T \Sigma_R^{-1} \\
E_L &= \Omega_L \Sigma_R^{-1} \\
E_R &= \Omega_R \Sigma_L^{-1} \\
D(y) &= E_L^{-1} K(y) \Sigma_R^{-1}
\end{aligned}
$$

As we will see, these quantities allow us to recursively compute the marginal probability of the dependency tree, $\hat{p}(x_1, \ldots, x_m)$, in a bottom up manner by using belief propagation.

To see this, let $hch(i)$ be the set of hidden children of hidden node i (in Figure 3.2 for instance, $hch(1) = \{2, 3\}$) and let $och(i)$ be the set of observed children of hidden node i (in the same figure $och(i) = \{1\}$). Then compute the marginal probability $\Pr(x_1, \ldots, x_m)$ from Equation 3.2 as

$$
r_i(h) = \prod_{j \in hch(i)} \alpha_j(h) \prod_{j \in och(i)} o(x_j|h) \tag{3.3}
$$

where $\alpha_i(h)$ is defined by summing over all the hidden random variables i.e., $\alpha_i(h) = \sum_{h'} \Pr(h'|h) r_i(h')$.

This can be written in a compact matrix form as

$$
\begin{aligned}
\overrightarrow{r_i}^\top = 1^\top & \prod_{j \in hch(i)} diag(T_{d_j}^\top \overrightarrow{r_j}) \\
& \cdot \prod_{j \in och(i)} diag(M^\top x_j)
\end{aligned} \tag{3.4}
$$

where $\overrightarrow{r_i}$ is a vector of size k (the dimensionality of the hidden space) of values $r_i(h)$. Note that since in Equation 3.2 we condition on whether x_j is the left or right child of its parent, we have separate transition matrices for left and right transitions from a given hidden node $d_j \in \{L, R\}$.

The recursive computation can be written in terms of observables as:

$$
\begin{aligned}
\overrightarrow{r_i}^\top = c_\infty^\top & \prod_{j \in hch(i)} D(E_{d_j}^\top \overrightarrow{r_j}) \\
& \cdot \prod_{j \in och(i)} D((U^\top U)^{-1} U^\top x_j)
\end{aligned}
$$

The final calculation for the marginal probability of a given sequence is

$$
\hat{\Pr}(x_1, \ldots, x_m) = \overrightarrow{r_1}^\top c_1 \tag{3.5}
$$

[7]The details of the derivation follow directly from the matrix versions of the variables.

The spectral estimation procedure is described below in Algorithm 1.

Algorithm 1 Spectral dependency parsing (Computing marginal probability of a tree.)

1: **Input:** Training examples- $x^{(i)}$ for $i \in \{1, \ldots, M\}$ along with dependency structures where each sequence $x^{(i)} = x_1^{(i)}, \ldots, x_{m_i}^{(i)}$.
2: Compute the spectral parameters $\hat{\mu}$, $\hat{\Sigma}_R$, $\hat{\Sigma}_L$, $\hat{\Omega}_R$, $\hat{\Omega}_L$, and $\hat{K}$
#Now, for a given sentence, we can recursively compute the following:
3: **for** $x_j^{(i)}$ for $j \in \{m_i, \ldots, 1\}$ **do**
4: Compute:

$$\overrightarrow{r_i}^\top = c_\infty^\top \prod_{j \in hch(i)} D(E_{d_j}^\top \overrightarrow{r_j}) \cdot \prod_{j \in och(i)} D((U^\top U)^{-1} U^\top x_j)$$

5: **end for**
6: Finally compute

$$\hat{\Pr}(x_1, \ldots, x_{m_i}) = \overrightarrow{r_1}^\top c_1$$

#The marginal probability of an entire tree.

3.3.4 Sample complexity

Our main theoretical result states that the above scheme for spectral estimation of marginal probabilities provides a guaranteed consistent estimation scheme for the marginal probabilities:

Theorem 2. *Let the sequence $\{x_1, \ldots, x_m\}$ be generated by an $k \geq 2$ state HMM. Suppose we are given a U which has the property that $U^\top M$ is invertible, and $|U_{ij}| \leq 1$. Suppose we use equation (3.5) to estimate the probability based on N independent triples. Then*

$$N \geq C_m \frac{k^2}{\epsilon^2} \log\left(\frac{k}{\delta}\right) \tag{3.6}$$

where C_m is specified in the appendix, implies that

$$1 - \epsilon \leq \left| \frac{\widehat{p}(x_1, \ldots, x_m)}{p(x_1, \ldots, x_m)} \right| \leq 1 + \epsilon$$

holds with probability at least $1 - \delta$.

Proof. A proof, in the case without directional transition parameters, can be found appendix A.4. The proof with directional transition parameters is almost identical. □

3.4 Experimental Evaluation

Since our algorithm can score any given tree structure by computing its marginal probability, a natural way to benchmark our parser would be to generate n-best dependency trees using some standard parser McDonald (2006); Koo et al. (2008) and then use our algorithm to re-rank the candidate dependency trees. One might expect purely generative models to not be competitive with state-of-the-art discriminative re-rankers Collins (2000); Koo and Collins (2005), however recently Sangati et al. (2009) have shown that carefully designed generative re-rankers can outperform discriminative re-rankers for Italian dependency parsing. So, we use a base parser McDonald (2006) to generate a set of n-best parses which we re-rank by computing their marginal probabilities as described in Algorithm 1.

Secondly, as a separate experiment we use our model to derive features for the MST parser based on the edge factored spectral log-probabilities.

3.4.1 Experimental Setup

Our base parser was the discriminatively trained MSTParser McDonald (2006), which implements both first and second order parsers and is trained using MIRA Crammer et al. (2006) and used the standard baseline features as described in McDonald (2006). We tested our methods on the English Penn Treebank Marcus et al. (1993). We use the standard splits of Penn Treebank; i.e., we used sections 2-21 for training, section 22 for development and section 23 for testing. We used the PennConverter[8] tool to convert Penn Treebank from constituent to dependency format. Following McDonald (2006); Koo et al. (2008), we used the POS tagger by Ratnaparkhi (1996) trained on the full training data to provide POS tags for development and test sets and used 10-way jackknifing to generate tags for the training set. As is common practice we stripped our sentences of all the punctuation. We evaluated our approach on sentences of all lengths.

3.4.2 Details of spectral learning

For the spectral learning phase, we need to just collect word counts from the training data as described above, so there are no tunable parameters as such. However,

[8] `http://nlp.cs.lth.se/software/treebank_converter/`

Method	Accuracy	Complete
I Order		
MST Parser	90.8	37.2
MST Parser+(RR, n=10)	91.1	37.4
MST Parser+(RR, n=50)	**91.3**	**37.6**
II Order		
MST Parser	91.8	40.6
MST Parser+(RR, n=10)	92.2	40.9
MST Parser+(RR, n=50)	**92.4**	**41.2**

Table 3.1: Dependency Parsing (Unlabeled) results for English test set (Section 23). **Note:** 1). n=10 or 50 indicates the number of parses which were re-ranked. 2). *Accuracy* is the number of words which correctly identified their parent and *Complete* is the number of sentences for which the entire dependency tree was correct.

we need to have access to an attribute dictionary U which contains a k dimensional representation for each word in the corpus. A possible way of generating U as suggested by Hsu et al. (2009) is by performing SVD on bigrams P_{21} and using the left eigenvectors as U. We instead used the eigenfeature dictionary proposed by Dhillon et al. (2011) (LR-MVL) which is obtained by performing CCA on neighboring words and has provably better sample complexity for rare words compared to the SVD alternative.

We induced the LR-MVL embeddings for words using the Reuters RCV1 corpus which contains about 63 million tokens in 3.3 million sentences and used their context oblivious embeddings as our estimate of U. We experimented with different choices of k (the size of the low dimensional projection) on the development set and found $k = 10$ to work reasonably well and fast. Using $k = 10$ we were able to estimate our spectral learning parameters $\mu, \Sigma_{L,R}, \Omega_{L,R}, K$ from the entire training data in under 2 minutes on a 64 bit Intel 2.4 Ghz processor.

3.4.3 Experiment 1: Re-ranking the outputs of MST parser

We used the MST parser to generate a list of n-best parses and we re-ranked them by their marginal probabilities as calculated using Algorithm 1. The results are shown in Table 3.1. As can be seen, the best results (corresponding to n=50) give up to 7.3% reduction in error over the baseline 1-best parser.

Method	Accuracy	Complete
I Order		
MST Parser	90.8	37.2
MST Parser(+S)	**91.1**	**37.4**
II Order		
MST Parser	91.8	40.6
MST Parser(+S)	**92.1**	**40.9**

Table 3.2: Dependency Parsing (Unlabeled) results for English test set (Section 23). 1) (+S) indicates using spectral features in addition to baseline features. 2) *Accuracy* is the number of words which correctly identified their parent and *Complete* is the number of sentences for which the entire dependency tree was correct.

3.4.4 Experiment 2: Spectral Features for a Discriminative parser

Next, in a separate experiment, we used our model to generate spectral features for use within the MST parser.

3.4.4.1 Spectral Features

Since the MST parser is edge factored, it can only use features over a single dependency attachment (or two attachments for a second order parser, where two of the words are the siblings on the same side of their parent). This poses a potential problem for using our spectral probabilities as calculated in Algorithm 1 as features, as they require recursive calculation over the entire tree structure.

To circumvent this difficulty, we compute the probability of a dependency attachment (first or second order), by doing the exact same probability calculation as described in Algorithm 1, but for each dependency attachment separately. I.e., we compute $\hat{p}(x_i, x_j)$, where x_i, x_j have a parent-child dependency relation. We repeatedly do this probability calculation for all first and second order dependency attachments and use their log-probabilities as additional features in the MST parser.

3.4.4.2 Discriminative Parsing Results

The results are shown in Table 3.2. Using the spectral features described in the previous section we got an unlabeled accuracy of 91.1% on the test set for first order parser and 92.1% for second order parser, and we get further gains by doing reranking in addition to adding spectral features. We believe that further improvement is possible by better tuning the features used.

3.5 Conclusion

In this paper we proposed a novel spectral method for dependency parsing. Unlike EM trained generative latent variable models, our method does not get stuck in local optima, it gives consistent parameter estimates, and it is extremely fast to train. We worked out the theory of a simple yet powerful generative model and showed how it can be learned using a spectral method. As pilot experimental evaluations we showed the efficacy of our approach by using the marginal probabilities output by our model as features in the MST parser as well as for re-ranking its outputs. Our method reduced the error of the baseline parser by up to 7.3%. Currently, we are working on building a stand-alone spectrally learnt generative parser which can estimate powerful dependency grammar models such as head automata Eisner (2000).

4

Using Regression to Estimate Parameters in Spectral HMM models

Hidden Markov Models (HMMs) are widely used to model discrete time series data, but the EM and Gibbs sampling methods used to estimate them are often slow or prone to get stuck in local minima. A more recent class of reduced-dimension spectral methods for estimating HMMs has attractive theoretical properties, but their finite sample size behavior has not been well characterized. We introduce a new spectral model for HMM estimation, a corresponding spectral bilinear regression model, and systematically compare them with a variety of competing simplified models, explaining when and why each method gives superior performance. Using regression to estimate HMMs has a number of advantages, allowing more powerful and flexible modeling.

4.1 Introduction

Hidden Markov Models (HMMs) Baum and Eagon (1967) are widely used in modeling time series data from text, speech, video and genomic sequences. In applications where the dimension of the observations is much larger than the dimension of the hidden state space, spectral methods can be used project the high dimensional observations down to a much lower dimensional representation that captures the information of the hidden state in the HMM. We call this class of model "spectral HMMs" (*sHMMs*) and show in this paper that sHMMs can be estimated in a variety of ways.

Standard algorithms for HMMs estimate the unobservable transition matrix T and emission matrix M, but are prone to getting stuck in local optima (for instance the EM algorithm) or are computationally intensive (Gibbs sampling). In contrast, sHMM methods estimate a fully observable representation of T and M and are fast, do not have local minima, have nice theoretical error bound proofs, and are optimal

Work from this chapter appears in Rodu et al. (2013)

in linear estimation sense.

Hsu et al. (2009) showed that a set of statistics using unigrams, bigrams and trigrams of observations are sufficient to estimate such models. We present a simpler estimation technique and show that it generalizes to a rich collection of regression-based methods for estimating HMMs. In regression, one can easily include more information such as a longer history, or more features about the observed data. These cannot as easily be added into a pure HMM model. Our methods are particularly useful for language modeling, where the emissions of the Hidden Markov Models are words drawn from a large vocabulary (tens or hundreds of thousands of words), and the hidden state is a much lower dimensional representation (30-100 dimensions).

HMMs of this size are widely used in modeling Natural Language Processing (NLP). Many variants of and applications of HMMs have been proposed including (to present a random list of recent work) multiple span-HMM to predict predicates in different domains Huang and Yates (2010), factorial-HMMs to resolve the pronoun anaphora Li et al. (2011), multi-chain HMMs to compute the meaning of terms in text Turdakov and Lizorkin (2009), tree-modified HMMs to do machine translation Zabokrtsky and Popel (2009), fertility-HMM to reduce word alignment errors Zhao and Gildea (2010) and continuous HMMs to summarize speech documents without text Maskey and Hirschberg (2006).

Our main HMM estimation method, which we call a *spectral HMM* is inspired by the observation in Hsu et al. (2009) that the "Observable Operator" model Jaeger (2000) which estimates the probability of a sequence $x_1, x_2, \ldots, x_t$ as

$$Pr(x_1, x_2, \ldots, x_t) = 1^\top A(x_t)A(x_{t-1}) \cdots A(x_1)\pi \tag{4.1}$$

in terms of the still unobservable $A(x) = T\text{diag}(M^\top x)$ (where $x = e_i$ denotes word i in a vocabulary, and e_i denotes as usual the vector of all zeros and a one in the i^{th} position) and the unigram probabilities π, can be rewritten to be a fully observable, partially reduced model through clever projections and combinations of the moment statistics. In chapter 2 we extend this to a fully reduced, fully observable model. This extension directly motivates simplified bilinear and regression estimation procedures.

We find that a wide range of spectral methods work well for estimating HMMs. HMMs have an intrinsically bi-linear model, but using a linear approximation works well in practice, especially when one sti recursive prediction. Our regression methods are competitive with the "traditional" method of moments methods, and make it relatively easy to add in much richer sets of features than either EM or standard spectral HMM estimations.

The rest of the chapter is organized as follows. In section 2 we formally describe the reduced dimension spectral HMM ($sHMM$) model and the bilinear and simplified regression models that it motivates. We also compare our $sHMM$ model against the partially reduced dimension model of Hsu et al. (2009). Section 3 gives our experimental results, and discusses prediction accuracy of the different methods in different limits. Section 4 concludes.

4.2 Approximations to HMMs

Consider a discrete HMM consisting of a sequence of observations $(x_1, x_2, ..., x_T)$ at discrete times $1...T$. Each observation, x_t is an indicator vector that corresponds to one of v labels (e.g. words). There is a corresponding sequence of hidden states, $(h_1, h_2, ..., h_T)$, where h_t corresponds to one of k labels.

Assume that $k << v$, as is the case, for example, when the vocabulary size v of words is much bigger than the hidden state size. Let T of size $k \times k$ denote the transition matrix; $T_{ij} = Pr(h_t = i|h_{t-1} = j)$. Let M of size $v \times k$ denote the emission matrix; $M_{ij} = Pr(x_t = e_i|h_t = j)$.

We estimate an sHMM using a matrix U which projects each observation x_t onto a low dimensional representation y_t using $y_t = U^\top x_t$, where x_t is defined as before. We work primarily in the y space, which is dimension k instead of the v-dimensional observation space. Note that unlike h, which is a discrete space, y lies in a continuous space.

U is the mapping between the original high dimension observation space and the reduced dimensional representation space. This matrix received a full treatment in Hsu et al. (2009) and therefore is not the focus of this paper. It is worth noting, however, that U is not unique, and need only satisfy a handful of properties. We call U the *eigenword* matrix, as $y = U^\top x$ forms a low dimensional representation of each word x in the vocabulary. For completeness, we note that a version of U can be easily estimated by taking the largest left singular vectors of the bigram matrix P_{21}, where

$$[P_{21}]_{i,j} = P(x_t = e_i, x_{t+1} = e_j).$$

We use this version in the empirical results presented below. This works well in theory (see details below) and adequately in practice, but better Us can be found, either by estimating U from another much bigger data set, or by using more complex estimation methods Dhillon et al. (2011).

In all of our methods, we will estimate a model to predict the probability of the next item in the sequence given what has been observed so far:

$$\Pr(x_{t+1}|x_t, x_{t-1}, \ldots, x_1) = \Pr(x_{t+1}, x_t, x_{t-1}, \ldots, x_1) / \Pr(x_t, x_{t-1}, \ldots, x_1).$$

We do this in the reduced dimension space of y_i.

4.2.1 sHMM Model and Estimation

Our core sHMM algorithm estimates $\Pr(x_t, x_{t-1}, \ldots, x_1)$ via the method of moments, writing it in terms of $c_\infty^\top$, c_1 and $C(y_t)$, and in turn writing each of these three items in terms of moments of the Ys. From Hsu et al. (2009) and chapter 2 we have

$$Pr(x_1, x_2, \ldots, x_t) = c_\infty^\top C(y_t) C(y_{t-1}) \cdots C(y_1) c_1 \tag{4.2}$$

with

$$c_1 = \mu, \quad c_\infty^\top = \mu^\top \Sigma^{-1}, \quad C(y) = K(y)\Sigma^{-1}$$

and parameters

$$\begin{aligned} \mu &= \boldsymbol{E}(y_1) = U^\top M\, \pi \\ \Sigma &= \boldsymbol{E}(y_2 y_1^\top) = U^\top M\, T\, \mathrm{diag}(\pi)\, M^\top U \\ K(a) &= \boldsymbol{E}(y_3 y_1^\top y_2^\top)a = U^\top M\, T\, \mathrm{diag}(M^\top U a)\, T\, \mathrm{diag}(\pi)\, (M^\top U) \end{aligned}$$

This yields the following estimate of Pr():

$$\widehat{\Pr}(x_t, x_{t-1}, \ldots, x_1) = \widehat{c}_\infty^\top \widehat{C}(y_t)\widehat{C}(y_{t-1}) \cdots \widehat{C}(y_1)\widehat{c}_1 \tag{4.3}$$

where

$$\widehat{c}_1 = \widehat{\mu}, \quad \widehat{c}_\infty^\top = \widehat{\mu}^\top \widehat{\Sigma}^{-1}, \quad \widehat{C}_y = \widehat{C}(y) = \widehat{K}(y)\widehat{\Sigma}^{-1}$$

and $\widehat{\mu}$, $\widehat{\Sigma}$ and $\widehat{K}()$ are the empirical estimates of the first, second and third moments of the Y's, namely

$$\widehat{\mu} = \frac{1}{N}\sum_{i=1}^{N} Y_{i,1}, \quad \widehat{\Sigma} = \frac{1}{N}\sum_{i=1}^{N} Y_{i,2}Y_{i,1}^\top, \quad \widehat{K}(y) = \frac{1}{N}\sum_{i=1}^{N} Y_{i,3}Y_{i,1}^\top Y_{i,2}^\top\, y$$

Here $Y_{i,t}$ indexes the N different independent observations (over i) of our data at time $t \in \{1, 2, 3\}$.

Our HMM model is shown in figure 4.1.

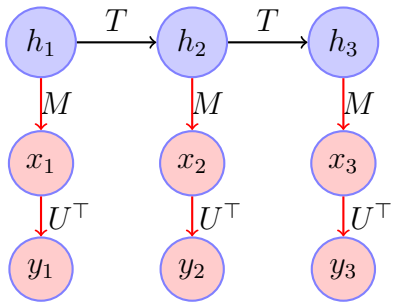

Figure 4.1: The HMM with states h_1, h_2, and h_3 which emit observations x_1, x_2, and x_3. These observations are further projected onto the lower dimensional space with observations y_1, y_2, y_3 by U from which our core statistic C_y is computed based on $K = E(y_3 \otimes y_1 \otimes y_2)$ which is a $(k \times k \times k)$ tensor

Chapter 2 proved that the *sHMM* model is PAC learnable if the true model is an HMM and the projection matrix U has the property that range$(M) \subset$ range(U) and $|U_{ij}| \leq 1$. Given any small error ϵ and small confidence parameter δ, when the sample

triples of observations are bigger than a polynomial of k, v, ϵ and δ, the probability estimated by reduced dimensional tensor $C(y)$ in Eqn. 4.3 is smaller than ϵ with high confidence $1-\delta$.

For any $t \in [2..\infty)$, the estimated value of y_t, denoted by $\hat{y}_t$, can be recursively estimated using the information at the previous time:

$$\hat{y}_t = \frac{C(y_{t-1})\hat{y}_{t-1}}{\hat{c}_\infty^\top C(y_{t-1})\hat{y}_{t-1}} \tag{4.4}$$

with $\hat{y}_1 = \widehat{\mu}$. Since the denominator in Eqn. 4.4 is a scalar constant for a particular time, we will separate the rescaling step from the recursive computation. Let $\omega_t = \hat{c}_\infty^\top C(y_{t-1})\hat{y}_{t-1}$. First we estimate $\tilde{y}_t = C(y_{t-1})\hat{y}_{t-1}$ using the information from time $t-1$, then we set $\hat{y}_t = \tilde{y}_t/\omega_t$.[1]

Note that once we have computed $\tilde{y}_t$, ω_t is computed deterministically; hence the key component in estimating $\hat{y}_t$ is the computation of

$$\tilde{y}_t = C(y_{t-1})\,\hat{y}_{t-1}. \tag{4.5}$$

The observable HMM representation with $\hat{y}_1$, $\hat{c}_\infty$ and $C(y)$ is sufficient to predict the probabilities of sequences of observations generated by an HMM. For joint probability of an observation sequence $(x_1, x_2, ..., x_t)$ one can use Eqn. 2.2. The conditional probability of the same sequence can be computed directly using $\hat{y}_t$. The conditional probability of observing i at time t is

$$Pr[x_t = e_i | x_1, x_2, ..., x_{t-1}] = [U\hat{y}_t]_i \tag{4.6}$$

This concludes the full presentation of the sHMM model. As mentioned in the introduction, this motivates simpler approximations which will now be discussed.

4.2.2 Bilinear Regression Model

Our *sHMM* model (4.5) that outputs the current $\tilde{y}_t$ is bilinear in y_{t-1} and $\hat{y}_{t-1}$. In other words, let $y_{j,t}$ be the j^{th} element of y_t, and $[\mathcal{C}]_{ijk} = c_{ijk}$. Then we can write

$$\tilde{y}_{j,t} \;=\; \sum_{i,k} c_{ijk} y_{i,t-1} \hat{y}_{k,t-1} \tag{4.7}$$

This leads naturally to our first simplified estimation technique–using linear regression by regressing $\tilde{y}_t$ on the outer product of y_{t-1} and $\hat{y}_{t-1}$ as shown in eqn. 4.7. We call this estimation method *Bilin-RRegr*. "Bilin"since it is Bilinear, "R" for recursive, since it is recursively estimated and predicted using the previous value of $\hat{y}_{k,t-1}$, and "Regr", since it is estimated using regression.

A note on training this model: in order to learn the parameters c_{ijk} we first

[1]Note the use of $\tilde{y}_t$ for the non-rescaled version of $\hat{y}_t$.

estimate $\tilde{y}$'s and $\hat{y}$'s using linear regression on our empirically collected trigram data using the actual $y = U^\top x$'s as the "responses" to be predicted. We then estimate the parameters in 4.7 using a second regression in which these initial estimates of y form the responses. One could iterate this to fixed point, but the above process is in practice sufficient.

Also, although *sHMM* uses the method of moments to estimate the parameters while *Bilin-RRegr* uses linear regression, when used to make predictions the two methods are used identically.

4.2.3 Other Regression Models:

As mentioned in the introduction, many methods can be used to estimate the $sHMM$ model. We focus on two main simplifications: one can linearize the bilinear model, and one can drop the recursive estimation. Recursion shows up in two places: when doing estimation, one can regress either on y_t and $\hat{y}_t$ or on y_t and y_{t-1}, and when using the model to predict, one can do a "rolling" prediction, in which y_{t+1} is predicted using the observed y_t and the predicted $\hat{y}_t$. These choices are made independently. For example the base spectral HMM method uses trigrams (no recursion) to estimate, but uses recursion to predict.

The bilinear equation in Eqn 4.7 can be linearized to give a simpler model to estimate $\tilde{y}_t$ using regression on y_{t-1} and $\hat{y}_{t-1}$. In the experimental results below, we call the resulting recursive linear model *Lin-RReg*:

$$\tilde{y}_t = \alpha\, y_{t-1} + \beta\, \hat{y}_{t-1} \tag{4.8}$$

We can also further simplify either the recursive bilinear model in Eqn 4.7 or the recursive linear model of Eqn 4.8 by noting that a simple linear estimate of $\hat{y}_{t-1}$ is $\hat{y}_{t-1} = Dy_{t-2}$. Since the matrix D is arbitrary, it can be folded into the model, giving a simple linear regression, *Lin-Regr*, model

$$\begin{aligned} \hat{y}_t &= \alpha\, y_{t-1} + \beta_1\, \hat{y}_{t-1} \\ &= \alpha\, y_{t-1} + \beta_2 A\, y_{t-2} \\ &= \alpha\, y_{t-1} + \beta\, y_{t-2} \end{aligned}$$

Note that here we estimate $\hat{y}$ directly instead of first estimating the unscaled $\tilde{y}$ and then rescaling to get our $\hat{y}$. Similarly, one can build a non-recursive bilinear model *Bilin-Regr*.

All of the above estimators work completely in the reduced dimension space Y. They are summarized in Table 1, along the single-lag version of *Lin-Regr*, *Lin-Regr-1*, and a couple of partially reduced dimension models which are described in the following section.

4.2.4 Partially Reduced Dimension Models

Instead of our fully dimension-reduced model $sHMM$, one can, following Hsu et al. (2009) estimate a tensor $B(x)$, which is only projected into the reduced dimension space in two of its three components. $B(x)$ thus takes an observation x, and produces an $k \times k$ matrix, unlike $C(y)$ which takes a reduced dimension y and produces an $k \times k$ matrix.[2]

Given $B(x)$, which is estimated from bigram and trigram occurrence counts, similarly to $C(y)$, the probability of the next item in a sequence is predicted using the same recursive (rolling) method described above. The fundamental equation is similar in form:

$$Pr(x_1, x_2, \ldots, x_t) = b_\infty^\top B(x_t)B(x_{t-1}) \cdots B(x_1)b_1 \tag{4.9}$$

See Hsu et al. (2009) for details. We call this method HKZ after its authors.

Our fully reduced dimension $sHMM$ offers several advantages over the original HKZ method detailed in chapter 2. Working entirely in the reduced dimension space reduces number of parameters to be estimated from k^2n to k^3. This comes at a cost in that the theorems for $sHMM$ require U to contain full range of M instead of only just being full dimension.

The other big change in this paper over Hsu et al. (2009) is the use of linear regression to estimate the model. Computing a regression, unlike using the method of moments, requires computing the inverse of the covariance of the features (the outer product of y_t and $\hat{y}_t$). At the cost of doing the matrix inversion, we get more accurate estimates, particularly for the rarer emissions.

Using a regression model also gives a tremendous increase in flexibility; The regression can easily include more terms of history, giving more accurate estimates, particularly for more slowly changing or non-Markovian processes. This comes at a cost of estimating more parameters, but if the history is included in linear, instead of a bilinear model, this is relatively cheap.

---

[2] Those familiar with the original paper will note that we have slightly re-interpreted B_x, which Hsu et al. call a matrix, and that what we call x here, they call δ_x.

Also the resulting $m \times m$ matrices are identical, specifically

$$\begin{aligned} C(y) &= K(y)\Sigma^{-1} \\ &= (U^\top M)T \operatorname{diag}(M^\top U y)(U^\top M)^{-1} \\ &= (U^\top M)T \operatorname{diag}(M^\top x)(U^\top M)^{-1} \\ &= B(x) \end{aligned}$$

Table 4.1: **Methods compared in our experiments.** "Num Params" is the number of parameters, not including the $k \times n$ parameters for U. $\hat{y}$ denotes the estimate of y scaled by ω_t as in Eqn. 4.4, and $\tilde{y}$ denotes the unscaled estimate

Method	Equation	Num. Params.
sHMM	$\tilde{y}_t = C(y_{t-1})\hat{y}_{t-1}$	k^3
Bilin-RRegr	$\tilde{y}_t = C(y_{t-1})\hat{y}_{t-1}$	k^3
Bilin-Regr	$\hat{y}_t = \Gamma(y_{t-1})y_{t-2}$	k^3
Lin-RRegr	$\tilde{y}_t = \alpha\, y_{t-1} + \beta\, \hat{y}_{t-1}$	$2k^2$
Lin-Regr	$\hat{y}_t = \alpha\, y_{t-1} + \beta\, y_{t-2}$	$2k^2$
Lin-Regr-1	$\hat{y}_t = \alpha\, y_{t-1}$	k^2
Lin-Regr-X	$\hat{x}_t = \alpha\, x_{t-1} + \beta\, x_{t-2}$	$2n^2$
HKZ	$\tilde{y}_t = B(x_{t-1})\hat{y}_{t-1}$	$k^2 n$
$EM_{(BaumWelch)}$	MLE	k^2

4.3 Experiments

In this section, we present experimental results on synthetic and real data for a variety of algorithms for estimating spectral HMMs.

Table (4.1) lists the methods we used in our experiments. The number of parameters being estimated in each case (not including the U projection matrix) are listed on the right side. We expect models with more parameters to better on larger training sets and worse on smaller ones.

4.3.1 Synthetic Data Test

The synthetic data is generated by constructing HMMs as follows: A potential transition matrix T is generated with elements from a folded normal distribution. It is accepted if its second eigenvalue is in the range 0.9 ± 0.1. Similarly, emission matrices M are generated with elements from a folded normal distribution and accepted if the second eigenvalue is in 0.8 ± 0.1. This allows us to generate a selection of HMMs, as well as to control the length of memory of the HMM and the difficulty of estimating it.

We run the experiments as follows. For each of 10 runs, we generate a random HMM model (T, M) as described above and use it to generate observation sequences from length $N = 100$ to $N = 1,000,000$ as training data and 100 short (length 10) sequences as test data.

We then estimate the various models using the training data. First we build the unigram P_1, bigram P_{21} and trigram P_{3x1} of the observations and use them to estimate the projection matrix U and model parameters such as α, β, Γ and C and B. U consists of the first k singular vectors corresponding to the k largest singular values of P_{21}. For the EM algorithm we use the R package (Himmelmann (2010)). Finally, we apply every method on each of test sequences and predict the last observation of

each test sequence given the preceding observations.

Each method in table (4.1) was tested varying several properties: training sequence lengths and the dimension of observations (figure 4.2), and the state transition probabilities (figure 4.3). In the last table, the second eigenvalue (2nd EV) of the transition matrix is varied. When this is close to 1, the process mixes slowly. In other words, it behaves close to a deterministic process. When this 2nd eigenvalue is close to zero, the process mixes rapidly. Basically it behaves like a sequence of IID hidden states. Hence more naive estimators will do well.

We report the prediction accuracy averaged over the 10 runs. We count a prediction as correct if the true observation has the highest estimated probability.

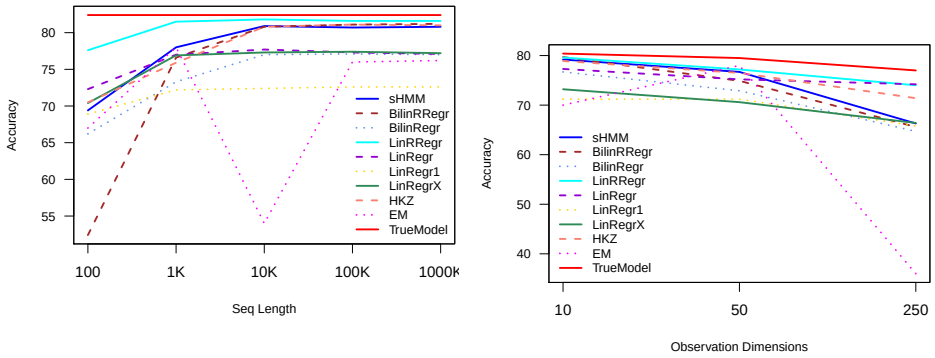

Figure 4.2: **Prediction accuracy on synthetic data.** Number of correct predictions of the 10th observation given the preceding 9 observations on 100 HMM sequences generated with dimension of states $m = 4$, second eigenvalue of transition matrix $T = 0.9$, second eigenvalue of emission matrix $M = 0.8$. Results are the average of 10 runs. The standard errors of 10 runs ranged from .06 to 3.1. **Left: Accuracy as a result of training sequence length. Observation dimension n = 10. Right: Accuracy as a result of observation dimension. Training length 10K**

4.3.2 NLP Data Test

We also evaluated our sHMM and rHMM on real NLP data sets. As with the synthetic data experiment, we predict the last word of a test sequence using the preceding words.

We use the New York Times Newswire Service (*nyt-eng*) portion of English Gigaword Fourth Edition corpus (LDC2009T13) in Penn Treebank Robert Parker (2009). We used a vocabulary of ten thousand words, including tokens for punctuation, sentence boundaries, and a single word token for all out-of-vocabulary words. The corpus consisted of approximately 1.3 billion words from 1.8 million document. Our training and test data set are drawn randomly without replacement from the *nyt-eng* corpus. The training data consists of long sequences of observations with lengths varying from 1K to 1000K. The test data consists of 10,000 sequences of observations of length 100.

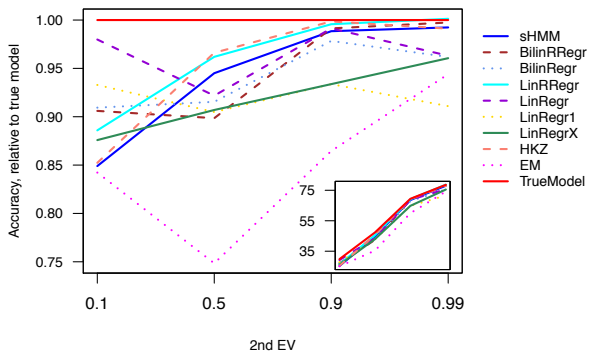

Figure 4.3: **Prediction accuracy relative to that of True Model on synthetic data in terms of the second eigenvalue of the transition matrix. Inset: actual prediction accuracy.** Number of correct predictions of the 10th observation given the preceding 9 observations on 100 HMM sequences. Results are the average of 10 runs. The standard errors of 10 runs ranged from 1.3 to 3.7. The model parameters are the number of states $m = 4$, the number of observations $n = 10$, the training length = 10K, and the second eigenvalue of the emission matrix $M = 0.8$

Following the language modeling literature, we use perplexity to measure how well our estimated language models fit the test data Brown et al. (1992); Rosenfeld (1996). Suppose a predicted distribution of a word x is p and the true distribution is q, the perplexity $PP(x)$ is defined as $PP(x) = 2^{H(p,q)}$, where $H(p,q)$ is the cross-entropy of p and q. i.e. $H(p,q) = -\sum_x q(x) \log_2 p(x)$. Because our true distribution q is a unit vector with only one element 1 at the x-th dimension, the actual computing of perplexity of word x is simplified as $PP(x) = \frac{1}{p_x}$. A lower perplexity $PP(x)$ indicates a better prediction on x.

We use the same test procedure and methods as for the synthetic data set. The perplexities of language models on *nyt-eng* corpus are shown in figure 4.4 with vocabularies of 1,000 and 10,000 words.

The results show several main trends, which are illustrated by two-way comparisons

- Fully reduced $sHMM$ vs. Partially reduced method HKZ
  - For small training sequences, $sHMM$ is better than HKZ, as one would expect, since $sHMM$ has far fewer parameters to estimate; $C(y)$ is m/n times smaller than $B(x)$. As theory predicts, in the limit of infinite training data, the two models are equivalent.
- Fully reduced $sHMM$ vs. Bilinear recursive regression *Bilin-RReg.*
  - On synthetic data generated from an HMM, for smaller training sets $sHMM$ performs better.
- Bilinear regression *Bilin-RReg.* vs. Linear regression *Lin-RReg.*

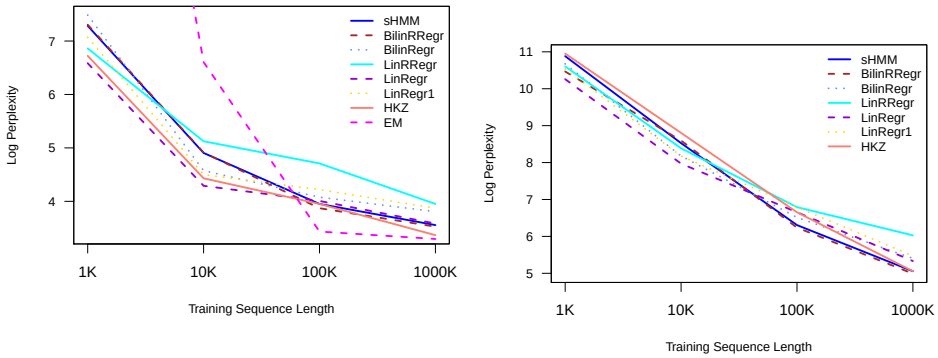

Figure 4.4: **Log of perplexities of language models on *nyteng* corpus. Left: corpus vocabulary size 1000 words. Right: corpus vocabulary size 10,000 words** Note: EM has been excluded on the right in order to preserve a sensible y-axis scale- the performance was poor across all sequence lengths

  - As expected, the simpler model linear model works better with short training sequences (We are not regularizing our regression, and so overfitting is possible). *Lin-RReg* unlike *Bilin-RReg*, is not a correct model of an HMM, and so will not perform as well in the limit of infinite training data.

- Recursive Methods (*Bilin-RReg*, *Lin-RReg*) vs. non-recursive ones (*Bilin-Reg*, *Lin-Reg*)

  - Recursive prediction always helps for linear models. For the more complicated bilinear model, recursion helps if there is sufficient training data. Keeping more lags in the model helps (e.g. *Lin-Regr* vs. *Lin-Regr1*).

- EM method *EM*

  - The *EM* method is prone to get stuck in local minima and often gives poor results. One could use a more sophisticated EM method, such as random restarts or annealing methods, but a major advantage of all of the spectral methods presented here is that they are fast (see, for example Cohen et al. (2013)) and guaranteed to converge to a good solution.

4.4 Discussion

HMM's are intrinsically nonlinear, but it is often advantageous to use a linear estimator, even when the data are generated by a model that is not linear. Linear estimators are fast, are guaranteed to converge to a single global optimum, and one can prove strong theorems about them. None of these properties are true of iterative algorithms such as EM for estimating nonlinear models.

We compared two major classes of techniques for estimating HMMs, method of moments ($sHMM$ and HKZ) and regression methods. All the methods presented here inherit the advantage of Hsu et al. (2009)'s method in that they use the projection matrix U containing the singular vectors of the bigram co-occurrence matrix to reduce the observations from a high dimension observation space X to a low dimension space Y. The Y space captures the information inherent in the hidden states and has same dimension as the hidden states. In this sense, the Y space can be seen as a linear transformation of the hidden state space. One could, of course, do regression in the original observation space, but that leads to models with vastly more parameters, making bilinear models prohibitively expensive. Models in the reduced dimension Y space have far fewer parameters and hence lower computational and sample complexity.

The method of moments models are simple to estimate, requiring only unigram bigram and trigram counts, and *not* requiring any recursive estimation (only recursive prediction). However, using regression models to estimate HMMs allows us far more flexibility than the method of moments models. Simple linear models can be used when training data are limited. Bilinear models that are identical to the $sHMM$ model can be used when more data are available. Longer histories can be used to estimate slowly changing HMMs (e.g. when the second eigenvalue of the transition matrix is close to 1) or when one does not believe that the HMM model is correct. Richer feature sets such as part of speech tags can also be added to the regression models when they are available.

Much work has been done generalizing the (partially reduced) HKZ method Siddiqi et al. (2010); Song et al. (2010) and extending it and our fully reduced $sHMM$ to probabilistic parsers Luque et al. (2012); Cohen et al. (2012, 2013). We believe that extensions of the regression-based estimators presented in this paper should prove valuable in these settings as well.

5

Spectral learning of continuous observation HMMs

Hidden Markov Models (HMMs) are widely used tools for modeling sequential data. Historically estimation of parameters in HMMs have relied on MLE methods such as Gibbs sampling and EM, though such methods suffer from slow run time and convergence concerns. Recently, a class of techniques, often called spectral techniques, have been developed that provide consistent, fully estimable ways of estimating key quantities in fully discrete HMMs through method of moments. We provide a novel algorithm for extending these spectral techniques to HMMs with output governed by continuous distributions.

5.1 Introduction

Hidden Markov models are an extremely useful class of models in which the latent state is assumed to be governed by a Markov process, and whose emissions at time t are emitted from a distribution conditional on the value of the hidden state at time t. Hidden Markov models encompass a wide variety of models, from models in discrete time, with a discrete hidden state space and discrete observations, to models with all continuous components, and any combination of discrete and continuous.

Hidden Markov models have enjoyed a wide range of applications. The fully discrete HMM is a major part of a toolkit in various aspects of language modeling, like document classification, word sense disambiguation, and part of speech tagging. Hidden Markov models with discrete time, discrete hidden state space, and continuous output distributions have had interest in applications such as speech recognition, robotics (for, say, gesture recognition for human/robot interactions or image recognition for robotic movement), and finance.

Recently a particular class of HMMs have become of interest due to the availability of, and increased ability to process, high-dimensional data. For instance we can model

Work from this chapter with Dean Foster and Lyle Ungar

the English language by assigning each word to an indicator vector in $\mathbb{R}^v$, where v is the size of the vocabulary. Speech processing, robotics, and finance each yield similarly high-dimensional, continuous data. Common to all of these domains is often a belief that the underlying process governing the data, the hidden state space is of a much lower dimension than the data itself. It is this class of HMMs that we will focus on in this paper.

Standard techniques for estimating these models rely on MLE methods such as Gibbs sampling and EM. These techniques often suffer in applications, however, due to their slow run time and risk of getting stuck in local optima. Recently, a technique often referred to in the literature as a spectral technique has arisen to estimate key components in fully discrete HMMs using method of moments. The spectral algorithm was originally proposed in Hsu et al. (2009) and was modified to a fully reduced dimensional form in chapter 2, a reduced rank form Siddiqi et al. (2010), and has seen extensions to other topologies like trees, as in Parikh et al. (2011) and chapter 3. Surprisingly, in many instances only three moments are needed to accurately estimate models.

In this paper we directly extend the "traditional" spectral HMM literature in that the parameters we estimate can be estimated from a low dimensional function of the training data, at the end of which the training data may be completely discarded. In the next section we present the extended, continuous output algorithm while simultaneously reminding the reader of the now widely used fully discrete version. The goal of this simultaneous presentation is two-fold. First, we hope to show the reader how this continuous version is a natural, powerful extension of the discrete case. Second, we hope to help build intuition about spectral estimation of HMM-type structures to enable easier extension to different hidden state space models.

Finally, one of the most important points from this work is that we are reducing the problem of estimation of a variety of HMM-type problems to the well-studied problem in machine learning and statistics of function estimation.

This is not the first attempt to apply spectral learning to continuous output HMMs. In Boots et al. (2010) the authors first embed the HMM into a reproducing kernel Hilbert space, then proceed to build theoretical observables that operate in a way similar to the observables of the traditional spectral HMM literature. It is not our aim in this paper to improve upon their algorithm. Rather we will provide a novel, more natural extension of the spectral literature to continuous output HMMs, and provide a much more flexible tool for solving a wide range of problems with a wide range of computational power and time. As previously stated, we do this by reducing the problem of spectral estimation of HMMs to a well known and highly flexible problem of functional estimation, thus putting flexibility into the researchers hands.

5.2 Hidden Markov Models and Notation

The reader is reminded that while we will discuss HMMs with both continuous and discrete output distributions, we will only work with HMMs in discrete time and with a discrete hidden state space. For clarity of presentation we will consider only HMMs whose latent dynamics are stationary, though this assumption can be easily relaxed.

We generally think of discrete observations (as well as the hidden states) as indicator vectors—so we represent the j^{th} observation (or state) by a vector of all 0's and a single 1 in the j^{th} entry. The dimensionality of the vector spaces is equal to the total number of possible observations.

Observations will be denoted by x in both discrete and continuous space, and will be subscripted by time. A key assumption about HMMs that allow spectral estimation is that observations live in a higher dimensional space than the hidden states. For instance, if there are k hidden states, and the observation lies in n dimensions (either there are n possible observations (i.e. vocabulary) or the observation comes from a distribution that emits n-dimensional observations), then $n > k$. For significant gains in efficiency we like n to be much bigger than k.

As mentioned before, the hidden state space is assumed to be governed by a Markov process, and we define a transition parameter over the hidden state space $T^{m \times m}$ such that $[T]_{ij} = \Pr(h_{t+1} = i | h_t = j)$. We further define a vector valued function $\lambda(x)$ which is a vector of conditional probabilities with the i^{th} entry being the probability of seeing observation x given hidden state i, $[\lambda(x)]_i = \Pr(x|h = i)$.

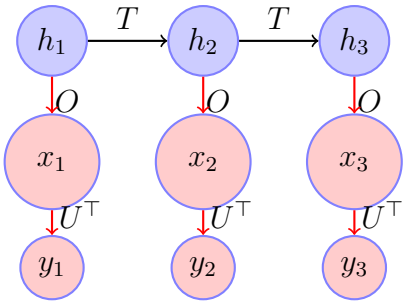

Figure 5.1: An HMM with states h_1, h_2, and h_3 which emit observations x_1, x_2, and x_3. The observations are further projected onto lower dimensional space with observations y_1, y_2, y_3 by U.

5.3 Observable Operators

Jaeger (2000) showed that HMMs can be represented as a subclass of an observable

operator model, which defines observable operators $A(x)$ such that

$$A(x) = T\text{diag}\lambda(x)$$

where $\text{diag}\lambda(x)$ is a diagonal matrix with $\lambda(x)$ on the diagonal. Setting $\pi_i = \Pr(h_1 = i)$, it is straightforward to verify that

$$\Pr(x_1, \ldots, x_t) = 1^\top A(x_t) \cdots A(x_1)\pi.$$

Note that if one had access directly to operators $A(x)$ one need not actually learn parameters T and $\lambda(x)$, however direct estimation of $A(x)$ is not possible, instead still requiring the direct estimation of the HMM parameters.

Spectral methods circumvent this problem by estimating a version of the observable operators that can be learned entirely from data directly.

5.4 Spectral Estimation

The hallmark of spectral estimation of hidden Markov Models is to estimate similarity transformations of the observable operator form of the HMM parameters. The similarity transform of $A(x)$ is often of the form

$$B(x) = (U^\top M)A(x)(U^\top M)^{-1}.$$

where M is a matrix such that the i^{th}, column of M is the expected value of X given hidden state i. Clearly, then, We have

$$\Pr(x_1, \ldots, x_t) = b_\infty^\top B(x_t) \cdots B(x_1) b_1$$

where $b_1 = U^\top M\pi$, and $b_\infty^\top = 1^\top (U^\top M)^{-1}$. Here it is worth mentioning that, for ease of presentation, we are considering systems of full rank—that is the matrix T is invertible, and, if M is $n \times k$, the column rank of M is k.

5.5 Building Observables

In this section we will describe how to build the observables $B(x)$. First note that the first three moments of the data from an HMM yield the following theoretical form, well known from the literature on spectral estimation of HMMs:

$$\begin{aligned} E[X_1] &= M\pi \\ E[X_2 \otimes X_1] &= MT\text{diag}\pi M^\top \\ E[X_3 \otimes X_1 \otimes X_2](\gamma) &= MT\text{diag}M^\top \gamma T\text{diag}\pi M^\top \end{aligned}$$

where we interpret $[a \otimes b \otimes c](d) = a \otimes b \cdot c^\top d$. This paper will be mostly concerned with moments 2 and 3, since they are instrumental in estimating $B(x)$. The nice thing about the intuitive extension of spectral estimation to the continuous output distribution case is that most all of the analysis from previous literature caries forward with almost no modification. Included in this is a discussion of the estimation of b_1 and b_∞, and as they are readily found in previous literature (and are really not the key objects of interest) we will not discuss them here.

We can already see from this presentation of the moments that it might be possible to isolate a similarity transformation of $A(x)$ by a combination of the third moment and the inverse of the second moment. However, the second moment lies in $\mathbb{R}^{m \times m}$ and is only of rank k. The key is to project these moments into a subspace in which they are full rank.

Consider an eigendictionary U (to be discussed later) such that $U^\top M$ is invertible, then estimating the second and third moments with reduced data $y = U^\top x$ allows

$$\begin{aligned} B(\gamma) &\equiv E[Y_3 \otimes Y_1 \otimes X_2](\gamma) E[Y_2 \otimes Y_1]^{-1} \\ &= (U^\top M) T \mathrm{diag} M^\top \gamma (U^\top M)^{-1} \end{aligned}$$

Recall that to this point we have not placed any assumption on our data being discrete or continuous. In either case, building observable $B(x)$ from the data naturally captures information about the hidden state space, namely the transition matrix.

To impress the notion that the hidden state dynamics are estimated separately from the observation parameters, algorithm 2 presents the algorithm for computing b_1, b_∞ and $B(\gamma)$ for both discrete observations and continuous observations.

Algorithm 2 Computing observables for spectral estimation of an HMM (fully reduced third moment from chapter 2)

1: **Input:** Training examples- $x^{(i)}$ for $i \in \{1, \ldots, M\}$ where $x^{(i)} = x_1^{(i)}, x_2^{(i)}, x_3^{(i)}$.
2: Compute $\hat{E}[x_2 \otimes x_1] = \frac{1}{M} \sum_{i=1}^m x_2^{(i)} x_1^{(i)\top}$.
3: Compute the left k eigenvectors corresponding to the top k eigenvalues of Σ. Call the matrix of these eigenvectors $\hat{U}$.
4: Reduce data: $\hat{y} = \hat{U}^\top x$.
5: Compute $\hat{m}u = \frac{1}{M} \sum_{i=1}^M y_1^{(i)}$, $\hat{\Sigma} = \frac{1}{M} \sum_{i=1}^M y_2^{(i)} y_1^{(i)\top}$ and tensor $\hat{C} = \frac{1}{M} \sum_{i=1}^M y_3^{(i)} \otimes y_1^{(i)} \otimes y_2^{(i)}$.
6: Set $\hat{b}_1 = \hat{\mu}$ and $b_\infty^\top = b_1^\top \hat{\Sigma}^{-1}$
7: Right multiply each slice of the tensor in the y_2 direction (so y_2 is being sliced up, leaving the $y_3 y_1^\top$ matrices intact) by $\hat{\Sigma}^{-1}$ to form $\hat{B}(\gamma) = \hat{C}(\gamma) \hat{\Sigma}^{-1}$

5.6 Estimating Observation Probabilities

5.6.1 Discrete Output

The other part of $A(X)$, namely $\text{diag}\lambda(x)$, is captured in the choice of γ, and how it interacts with $M^\top$. Recall that the i^{th} column of M is the expected value of x given hidden state i. Now, in the discrete case (where the j^{th} observation is represented by a vector of 0's with a single 1 in the j^{th} entry), the expected vector is precisely a probability vector such that the j^{th} entry of the expected vector in the i^{th} column is exactly $\Pr(x_j|h_i)$. Therefore, the j^{th} row of M is $\lambda(x_j)$.

This is the formulation of HKZ, so the choice for γ at time t is simply x_t, or the observation at time t, as $M^\top x_t$ extracts the row corresponding to the observation at time j.

One extension of Hsu et al. (2009), chapter 2, shows it is possible to estimate a fully reduced third moment, $E[Y_3 \otimes Y_1 \otimes Y_2](\gamma)$. In terms of the theoretical quantities, the first two moments remain the same, while the third becomes

$$E[Y_3 \otimes Y_1 \otimes Y_2](\gamma) = \\ U^\top M T \text{diag} M^\top U \gamma T \text{diag} \pi M^\top U$$

They show that the appropriate choice for γ in that case is y.

At this point we will describe the choice of γ necessary for the continuous output HMM. Following RDHMM, we will use the fully reduced version of the third moment matrix.

5.6.2 Continuous Output

Define $g(x) \equiv E[Y_2|x_1]$. First, we build intuition for the theoretical form of $g(x)$ given the observations are generated from an HMM. Let $\tilde{h}_t$ be the probability vector associated with begin in a particular state at time t. Then

$$E[y_2|\tilde{h}_2] = U^\top M \tilde{h}_2.$$

Also,

$$E[h_2|\tilde{h}_1] = T\tilde{h}_1.$$

thus $E[y_2|h_1] = U^\top M T \tilde{h}_1$. To establish a belief about h_1 given x_1, recall from Bayes formula

$$\Pr(h_1|x_1) = \frac{\Pr(x_1|h_1)\Pr(h_1)}{\Pr(x_1)}$$

We can arrange each probability into a vector, and because in the indicator vector case the probability vector is the same as the expected value vector, we have, in vector notation

$$E[h_1|x_1] = \frac{\text{diag}\pi\lambda(x)}{\pi^\top\lambda(x)}$$

and so putting together the pieces we get

$$E[y_2|x_1] = \frac{U^\top MT\text{diag}\pi\lambda(x)}{\pi^\top\lambda(x)}$$

Recall that the goal is to isolate $\lambda(x)$. Note that

$$\begin{aligned} E[y_2 \otimes y_1]^{-1} g(x) &= \frac{(M^\top U)^{-1}\lambda(x)}{\pi^\top\lambda(x)} \\ &\equiv G(x) \end{aligned}$$

When this is plugged into our fully reduced version of $B(x)$, we get

$$\begin{aligned} B(G(x)) &= (U^\top M)T\text{diag}M^\top UG(x)(U^\top M)^{-1} \\ &= (U^\top M)T\text{diag}\lambda(x)(U^\top M)^{-1}\frac{1}{\Pr(x)} \end{aligned}$$

where $\Pr(x)$ is the marginal probability. $B(G(x))$ is exactly what we want, up to a constant factor depending on x. This condition also appears in the RKHS version.

5.7 Sample Complexity

As we are primarily concerned with extending the model of the discrete output spectral HMM to the continuous version, so too will we extend the typical sample complexity theorems from the discrete to the continuous version. The proof techniques are the same, especially from chapter 2. We are in particular interested in bounding the relative probability of a sequence of observations. We need one

Theorem 3. *Let X_t be generated by an $k \geq 2$ state HMM. Suppose we are given a U which has the property that range$(M) \subset$ range(U) and $|U_{ij}| \leq 1$, a function $g(x)$ such that the relative error of $g(x)$ is less than α. Suppose we use equation (4.3) to estimate the probability based on N independent triples. Then*

$$N \geq \frac{128k^2}{(\sqrt[3t+3]{1+\epsilon} - 1)^2\ \Lambda^2\sigma_k^4} \log\left(\frac{2k}{\delta}\right) \tag{5.1}$$

implies that

$$\beta_0 \leq \left| \frac{\widehat{\Pr}(x_1, \ldots, x_t)}{\Pr(x_1, \ldots, x_t)} \right| \leq \beta_1$$

holds with probability at least $1-\delta$*, where* $\beta_0 = (1-\epsilon)(1-\alpha)^t$ *and* $\beta_1 = (1+\epsilon)(1+\alpha)^t$.

Note that the upper and lower bounds of the sample complexity can be neatly broken down into two components—error from estimation of $g(x)$, and error from estimation of the observable quantities. This parallels nicely the breakdown of estimation of the hidden state dynamics through the observable moments and the estimation of the observation parameters through the function $g(x)$. Note that we assume that the g function is estimated separately from the other parameters, and we consider this a fixed cost- in other words increasing the sample size associated with this theorem will not improve the relative error of the g function. Therefore, for the proofs below we assume the user has total information about the g function, and hence it contributes no error.

The maximal relative error for any given term contains Λ, the smallest value in any of the terms of the parameters, in the denominator. Because this is never observed, we recast the probability calculations using the smallest observed value. This is possible given that we bound the relative error for any possible term in the observables, and we state a final corollary that recasts the theorem with this consideration.

The proof proceeds in a series of lemmas. For clarity we will simply state all the lemmas necessary to prove the main theorem, providing their proofs in the Appendix, with the exception of the final lemma that implies the theorem itself.

Lemma 4. *Our estimates of all elements of* μ*,* Σ^{-1} *and* $K()$ *are bounded by* $3J/\sigma_k^2$ *with probability* $1-\delta$*, where* $J \equiv 2k\sqrt{\frac{2\log\frac{2k}{\delta}}{N}}$.

Using lemma 4 we obtain the following lemma.

Lemma 5. *From the method of moments estimate of the parameters, we obtain the following bound*

$$\left(1 - \frac{3J}{\sigma_k^2 \Lambda}\right)^{3t+3} (1-\alpha)^t \leq \frac{\widehat{\Pr}(x_1, \ldots, x_t)}{\Pr(x_1, \ldots, x_t)} \leq \left(1 + \frac{3J}{\sigma_k^2 \Lambda}\right)^{3t+3} (1+\alpha)^t$$

The next lemma converts the sample complexity into more useful bounds on Λ and σ_k.

Lemma 6. *If*

$$N \geq \frac{128k^2}{(\sqrt[3t+3]{1+\epsilon} - 1)^2 \Lambda^2 \sigma_k^4} \log\left(\frac{2k}{\delta}\right)$$

then

$$\Lambda \geq \frac{3J}{\sigma_k^2(\sqrt[3t+3]{1+\epsilon} - 1)} \tag{5.2}$$

$$\sigma_k \;\geq\; 4J \tag{5.3}$$

Finally we can use these bounds to get:

Lemma 7. *If equation (5.1) of Theorem 3 is replaced by (5.2) and (5.3) then the results of the theorem follow.*

We prove this lemma, and hence the theorem, now.

Proof of lemma 7: Note from lemma (6) we have

$$\begin{aligned}\Lambda &\geq \frac{3J}{\sigma_k^2(\sqrt[3t+3]{1+\epsilon}-1)} \\ &\geq \frac{3J}{\sigma_k^2(1-\sqrt[3t+3]{1-\epsilon})}\end{aligned}$$

First working with the lower bound from lemma (5), we have

$$\begin{aligned}\left(1-\frac{3J}{\sigma_k^2\Lambda}\right)^{3t+3}(1-\alpha)^t &\geq \left(1-\frac{3J}{\sigma_k^2\frac{3J}{\sigma_k^2(1-\sqrt[3t+3]{1-\epsilon})}}\right)^{3t+3}(1-\alpha)^t \\ &\geq \left(1-(1-\sqrt[3t+3]{1-\epsilon})\right)^{3t+3}(1-\alpha)^t \\ &\geq \left(\sqrt[3t+3]{1-\epsilon})\right)^{3t+3}(1-\alpha)^t \\ &\geq (1-\epsilon)\,(1-\alpha)^t\end{aligned}$$

And working with the upper bound we have

$$\begin{aligned}\left(1+\frac{3J}{\sigma_k^2\Lambda}\right)^{3t+3}(1+\alpha)^t &\leq \left(1+\frac{3J}{\sigma_k^2\frac{3J}{\sigma_k^2(\sqrt[3t+3]{1+\epsilon}-1)}}\right)^{3t+3}(1+\alpha)^t \\ &\leq \left(1+\sqrt[3t+3]{1+\epsilon}-1\right)^{3t+3}(1+\alpha)^t \\ &\leq \left(\sqrt[3t+3]{1+\epsilon}\right)^{3t+3}(1+\alpha)^t \\ &\leq (1+\epsilon)\,(1+\alpha)^t\end{aligned}$$

As mentioned before, the sample complexity bound in Theorem 3 relies on knowing unobserved parameters of the problem. To avoid this, we modify Lemma 7 to make it observable. In other words, we convert the assumptions of sample complexity into a checkable condition.

Corollary 4. *Let X_t be generated by an $k \geq 2$ state HMM. Suppose we are given a U which has the property that range$(O) \subset$ range(U). Suppose we use equation (4.3) to estimate the probability based on N independent triples. Then with probability $1-\delta$,*

if the following two inequalities hold

$$\widehat{\Lambda}\,\widehat{\sigma}_k^2 \geq \left(12k + \frac{6m}{(\sqrt[3t+3]{1+\epsilon}-1)}\right)\sqrt{\frac{2\log\frac{2k}{\delta}}{N}} \tag{5.4}$$

$$\widehat{\sigma}_m \;\geq\; 10m\sqrt{\frac{2\log\frac{2k}{\delta}}{N}}. \tag{5.5}$$

then

$$1-\epsilon \leq \left|\frac{\widehat{P}(x_1,\ldots,x_t)}{P(x_1,\ldots,x_t)}\right| \leq 1+\epsilon$$

The advantage of the corollary is that the left hand sides of the two conditions are observable and the right hand sides involve known quantities. Hence one can tell if the condition is true or not–it doesn't require knowing unobserved parameters. Note that the statement is of the form $\Pr(A \Rightarrow B) \geq 1-\delta$ so interpretation must be done carefully.

To prove corollary (4) we need two technical lemmas: Lemma 8 and Lemma 9. They are stated and proved in appendix A.2. Lemma 8 basically says that with high probability, each element of μ, Σ and $K()$ is estimated accurately. This is then used in Lemma 9 to show that Λ and σ_k are estimated accurately.

5.8 Conclusion

We have presented a spectral method for estimating HMMs with a continuous-valued observation space. We show that the algorithm proceeds by estimating the usual $C()$ tensor from 2 and by further estimating a new function $g()$, and that these two functions (as they can be seen) factor the problem of estimating HMMs with spectral methods into estimation of the hidden state space, and estimation of the observation space. Both functions can be estimated in highly flexible ways and incorporate a variety of information beyond simply the observations of interest.

6

Conclusions and Future Directions

We now have a framework in which to think about spectral estimation of HMMs and HMM-like objects. There are two immediate directions in which to extend the work in this thesis. We would like to improve the sample complexity for estimating the observables, as well as extend spectral estimation to cover a broader class of models.

In this thesis we focused on discrete-time, discrete hiddden state space models. In addition, beside the extension to a tree topology, this thesis considered linear chain hidden state dynamics. However, HMMs can be arbitrarily generalized to a wide range of models. We can extend the hidden state space to be continuous vectors instead of discrete vectors (the Kalman filter estimates a model of this type) and can extend the time granularity to be continuous.

Further, the hidden state space can itself be factored. For instance, some models consider the case in which there are some number of concurrent hidden state chains that have some dependency structure placed on them (allowing arbitrary dependence structure does not provide any gains in efficiency). Others have layered hidden state chains- for instance a slow moving hidden state chain that governs (or operates independently from) a faster moving hidden state chain (NLP provides an intuitive example for this type of model, in which topics relate to the slow moving hidden state space, and grammar relates to the faster moving hidden state space).

One focus of future research is to extend spectral methods to these classes of latent state models. The hope is that with the fully factored framework that we developed for spectral estimation here, we can easily extend spectral estimation to apply to these other modeling possibilities.

Regarding the sample complexity, the proof technology used in 2 requires that the sample complexity depend on Λ which, though it's empirical version is both fully observable and has a known relationship to the theoretical quantity, is entirely uncontrollable and often explodes the sample complexity. Further, there is no intuitive reason why the sample complexity really depends on Λ- it is simply an artifact of the proof. In practice, the sample complexity of the spectral method in chapter 2 is much smaller than the bound would indicate. We hope to find a better technology to obtain a sample complexity or confidence statement about the method presented in chapter 2 that better reflects the true sample complexity.

A

Appendices

A.1 Notation

We use the following notation throughout this document:

- v - dimensionality of the observation space
- k - dimensionality of the hidden state space
- x - an observation, discrete or continuous, in unreduced form
- y - the reduced dimensional form of x, where $y = U^\top x$.
- T - $k \times k$ transition matrix
- M - $v \times k$ matrix in which the columns are the expected value of observations given the hidden state
- $\lambda(x)$ - a k dimensional vector giving the probability of observation x given the hidden state
- U - the eigendictionary

- $C()$ - a tensor, or function, that takes a vector and returns a matrix that is the observable operator for the observation x that maps to the reduced dimensional observation y
- Σ - $E[y_2 \otimes y_1]$
- K - $E[y_3 \otimes y_1 \otimes y_2]$
- σ_m - The m^{th} smallest singular value of Σ.
- Λ - $\min\{\min_i |\mu_i|, \min_{i,j} |\Sigma^{-1}_{ij}|, \min_{i,j,l} |K_{ijl}|\}$

A.2 Supplement for Chapter 2

Lemma 1 (**Restatement of Lemma 1).** *Assume the hidden state is of dimension k and the rank of M is also k. Then:*

$$\Pr(x_1, x_2, \ldots, x_t) = 1^\top A(x_t)A(x_{t-1}) \cdots A(x_1)\pi \tag{2.1}$$

$$\Pr(x_1, x_2, \ldots, x_t) = b_\infty^\top B(x_t)B(x_{t-1}) \cdots B(x_1)b_1 \tag{2.5}$$

$$\Pr(x_1, x_2, \ldots, x_t) = c_\infty^\top C(y_t)C(y_{t-1}) \cdots C(y_1)c_1 \tag{2.6}$$

Where (2.5) requires $U^\top M$ to be invertible, and (2.6) requires range$(M) \subset$ range(U).

Proof:

As pointed out in the main text, Jaeger (2000) showed (2.1), and Hsu et al. (2009) showed (2.5). To show (2.6), we will first write the characteristics μ, Σ and K in terms of the theoretical matrices, T, M, U, and π:

$$\mu \;=\; U^\top M\pi$$

$$
\begin{aligned}
\Sigma &= U^\top M\, T\, \mathrm{diag}(\pi)\, M^\top U \\
\Sigma^{-1} &= (M^\top U)^{-1}\, \mathrm{diag}(\pi)^{-1}\, T^{-1}\, (U^\top M)^{-1} \\
K(y) &= U^\top M\, T\, \mathrm{diag}(M^\top U y)\, T\, \mathrm{diag}(\pi)\, M^\top U
\end{aligned}
$$

By definition, we have

$$
c_1 \equiv \mu = U^\top M\, \pi
$$

likewise,

$$
\begin{aligned}
c_\infty^\top &\equiv \mu^\top \Sigma^{-1} \\
&= \left(\pi^\top M^\top U\right) \left((M^\top U)^{-1}\, \mathrm{diag}(\pi)^{-1}\right. \\
&\qquad\qquad \left. \cdot\, T^{-1}\, (U^\top M)^{-1}\right) \\
&= \pi^\top\, \mathrm{diag}(\pi)^{-1}\, T^{-1}\, (U^\top M)^{-1} \\
&= \mathbf{1}^\top T^{-1} (U^\top M)^{-1} \\
&= \mathbf{1}^\top (U^\top M)^{-1}
\end{aligned}
$$

For C,

$$
\begin{aligned}
C(y) &= K(y)\, \Sigma^{-1} \\
&= U^\top M\, T\, \mathrm{diag}(M^\top U y) \\
&\qquad\qquad \cdot\, T\, \mathrm{diag}(\pi)\, M^\top U\, \Sigma^{-1} \\
&= U^\top M\, T\, \mathrm{diag}(M^\top U y) (U^\top M)^{-1}
\end{aligned}
$$

Note that $UU^\top$ is a projection operator and since its range is the same as that of M

we have $M^\top UU^\top = M^\top$. So, if $y = U^\top x$, then:

$$\begin{aligned} C(y) &= U^\top M\ T\ \mathrm{diag}(M^\top UU^\top x)(U^\top M)^{-1} \\ &= U^\top M\ T\ \mathrm{diag}(M^\top x)(U^\top M)^{-1} \\ &= U^\top M\ A(x)\ (U^\top M)^{-1} \end{aligned}$$

Thus (2.6) follows from a telescoping product.

□

Proof of lemma 6:

The proof is simply algebraic manipulation. We have

$$N \geq \frac{128k^2}{(\sqrt[2t+3]{1+\epsilon} - 1)^2 \Lambda^2 \sigma_k^4} \log\left(\frac{2k}{\delta}\right)$$

which implies that

$$\begin{aligned} \Lambda^2 &\geq \frac{128k^2}{(\sqrt[2t+3]{1+\epsilon} - 1)^2 N \sigma_k^4} \log\left(\frac{2k}{\delta}\right) \\ &\geq \frac{72k^2}{(\sqrt[2t+3]{1+\epsilon} - 1)^2 N \sigma_k^4} \log\left(\frac{2k}{\delta}\right) \end{aligned}$$

and taking the square root and making the relevant substitution for J we have

$$\Lambda \geq \frac{3J}{\sigma_k^2(\sqrt[2t+3]{1+\epsilon} - 1)}$$

To show the bound for σ_k we have that

$$N \geq \frac{128k^2}{(\sqrt[2t+3]{1+\epsilon} - 1)^2 \Lambda^2 \sigma_k^4} \log\left(\frac{2k}{\delta}\right)$$

and noting that $\Lambda < 1$ and $\sqrt[2t+3]{1+\epsilon} - 1 < 1$,

$$\sigma^4 \geq \frac{128k^2}{N} \log\left(\frac{2k}{\delta}\right)$$

Taking the square root of both sides and making the relevant substitution, we get

$$\sigma_k^2 \geq 4J$$

and since $\sigma_k < 1$ implies $\sigma_k^2 < \sigma_k$ then we get the desired inequality. □

Lemma 8. *Our estimates of all elements of* μ, Σ^{-1} *and* $K()$ *are bounded by* $3J/\sigma_k^2$ *with probability* $1-\delta$, *where* $J \equiv 2k\sqrt{\frac{2\log\frac{2k}{\delta}}{N}}$.

Proof:

We first derive absolute bounds for each entry of μ, Σ and $K()$. To handle all three of them at the same time, we will generically call any one of these three "θ" and its estimate $\widehat{\theta}$. Suppose that $\widehat{\theta}$ has g entries that are taking the mean with N observations all of which are bounded between -1 and 1. Then, for each entry we have from Hoeffding (1963) that

$$\Pr(|\widehat{\theta}_i - \theta_i| > \epsilon) \leq 2e^{\frac{-N\epsilon^2}{2}}$$

and so

$$\Pr(\exists\ i \text{ s.t. } |\widehat{\theta}_i - \theta_i| > \epsilon) \leq 2ge^{\frac{-N\epsilon^2}{2}}$$

and setting $2ge^{\frac{-N\epsilon^2}{2}} = \delta$ we solve that $\epsilon = \sqrt{\frac{2\log\frac{2g}{\delta}}{N}}$ so with probability $1-\delta$ we have that

$$\forall i \quad |\widehat{\theta}_i - \theta_i| \leq \sqrt{\frac{2\log\frac{2g}{\delta}}{N}}.$$

Note that for μ, Σ and $K()$ we have a vector, a matrix and a tensor that are estimated as $E(Y_1)$, $E(Y_1 Y_2^\top)$ and $E(Y_3 Y_1^\top Y_2^\top)$ respectively with k, k^2 and k^3 entries respectively, we see that the total number of entries in all three of them is less than k^4. (Except in the trivial case where $k = 1$. But this corresponds to the data being IID and so doesn't count as a HMM.) So all three of the following hold simultaneously with probability $1 - \delta$:

$$\begin{aligned} \forall i \quad |\widehat{\mu}_i - \mu_i| &\leq \sqrt{\frac{8 \log \frac{2k}{\delta}}{N}} \\ \forall i, j \quad |\widehat{\Sigma}_{ij} - \Sigma_{ij}| &\leq \sqrt{\frac{8 \log \frac{2k}{\delta}}{N}} \\ \forall i, j, j \quad |[\widehat{K}]_{ijl} - [K]_{ijl}| &\leq \sqrt{\frac{8 \log \frac{2k}{\delta}}{N}} \end{aligned} \tag{A.1}$$

Lastly we need to bound Σ^{-1}. We will start by bounding the norm of $\widehat{\Sigma} - \Sigma$. By (A.1) we see $||\widehat{\Sigma} - \Sigma||_{\max} \leq \sqrt{\frac{8 \log \frac{2k}{\delta}}{N}}$, by the relationship $||M||_2 \leq m||M||_{\max}$ for $k \times k$ square matrices, we get the desired result.

From this bound on $||\widehat{\Sigma} - \Sigma||_2$ and lemma 20 of Hsu et al. (2009) we have that

$$|\widehat{\sigma}_k - \sigma_k| \leq J \tag{A.2}$$

where σ_k is the smallest singular value for Σ. By their Lemma 23 we then have that

$$||\widehat{\Sigma}^{-1} - \Sigma^{-1}||_2 \leq \frac{1 + \sqrt{5}}{2} \left(\frac{1}{\widehat{\sigma}_k - J} \right)^2 J$$

By assumption $\sigma_k > 4J$, we see $\sigma_k - J > 3\sigma_k/4$. Thus from the algebra that $\frac{1+\sqrt{5}}{2}(\frac{4}{3})^2 \leq 3$, we see

$$||\widehat{\Sigma}^{-1} - \Sigma^{-1}||_2 \leq 3J/\sigma_k^2.$$

From $||\widehat{\Sigma}^{-1} - \Sigma^{-1}||_{\max} \leq ||\widehat{\Sigma}^{-1} - \Sigma^{-1}||_2$ we get our element-wise norm on the errors. Since $\sigma_k \leq 1$, we see that

$$3J/\sigma_k^2 \geq 3J = 3k\sqrt{\frac{8\log\frac{2k}{\delta}}{N}} \geq \sqrt{\frac{8\log\frac{2k}{\delta}}{N}}$$

□

Lemma 9. *The estimates of Λ and σ_k have the following accuracy:*

$$\begin{aligned} |\widehat{\Lambda} - \Lambda| &\leq \frac{6k}{\sigma_k^2}\sqrt{\frac{2\log\frac{2k}{\delta}}{N}} \\ |\widehat{\sigma}_k - \sigma_k| &\leq 2k\sqrt{\frac{2\log\frac{2k}{\delta}}{N}}. \end{aligned}$$

with probability greater than $1 - \delta$.

Proof: $\widehat{\Lambda}$ is the empirical minimum of all the

$$\widehat{\Lambda} \equiv \min\{\min_i |\widehat{\mu}_i|, \min_{i,j} |\widehat{\Sigma}_{ij}^{-1}|, \min_{i,j,l} |\widehat{K}_{i,j,l}|\}$$

From lemma 8 we have bounded the accuracy of the estimate of each element of μ, Σ and $K()$, the minimum of these will be estimated within the same accuracy. This established (A.3).

The second inequality (A.3) was also established in the proof of the theorem in equation (A.2).

□

A.3 Likelihood ratio version of theorem 3

In 2.3.1 we considered the likelihood ratio as a way of getting a better estimator. There we used a weighting vector p_i which normalized our probability. In other words,

$$\frac{\Pr(x_1, x_2, \ldots, x_t)}{p_{x_1} p_{x_2} \cdots p_{x_t}}$$

It will be a bit more mathematically convenient if we instead use $q_i = 1/\sqrt{p_i}$ instead. So, define:

$$Q(x_{1:t}) = Q(x_1, x_2, \ldots, x_t) = q(x_1)q(x_2)\cdots q(x_t)$$

Then our "likelihood ratio" is

$$\lambda(x_1, x_2, \ldots, x_t) = \Pr(x_1, x_2, \ldots, x_t)Q(x_1, x_2, \ldots, x_t)^2$$

We will think of these q_i's as a vector and define

$$M^* \equiv \mathrm{diag}(q)M$$

and

$$A^*_x \equiv T\mathrm{diag}(M^{*T}\mathrm{diag}(q)x)$$

We will then be able to show a similar product rule as (2.1):

$$\Pr(x_{1:t})Q^2(x_{1:t}) = 1^\top A^*(x_t)A^*(x_{t-1})\cdots A^*(x_1)\pi.$$

The version of this product rule we will estimate is also similar. We will define $U^* = \mathrm{diag}(q)U$ and $y^*_t = U^{*\top}\mathrm{diag}(q)x_t = U^\top \mathrm{diag}(q)^2 x_t$. Our statistics are then:

$$\mu^* \equiv E(y^*_1)$$

$$\begin{aligned}\Sigma^* &\equiv E(y_2^* y_1^{*\top}) \\ K^*(a) &\equiv E(y_3^* y_1^{*\top} y_2^{*\top})a\end{aligned}$$

Defining our characteristics as before:

$$\begin{aligned}c_1^* &\equiv \mu^* \\ c_\infty^{*\top} &\equiv \mu^{*\top}\Sigma^{*-1} \\ C^*(y^*) &= K^*(y^*)\,\Sigma^{*-1}\end{aligned}$$

These can also be used to estimate λ as the following lemma shows:

Lemma 10. *Assume the hidden state is of dimension k and the rank of M is also k. Then:*

$$\begin{aligned}\lambda(x_1,\ldots,x_t) &\equiv \Pr(x_{1:t})Q^2(x_{1:t}) \\ &= 1^\top A^*(x_t)A^*(x_{t-1})\cdots A^*(x_1)\pi \\ &= c_\infty^{*\top} C^*(y_t^*)\cdots C^*(y_1^*)c_1^* \qquad \text{(A.3)}\end{aligned}$$

Where the last equation requires
range$(M) \subset$ *range*$(U\mathrm{diag}(q))$.

Proof:

$$\begin{aligned}A_x^* &\equiv T\ \mathrm{diag}(M^{*\top}\mathrm{diag}(q)x) \\ &= T\ \mathrm{diag}((\mathrm{diag}(q)M)^\top \mathrm{diag}(q)x) \\ &= T\ \mathrm{diag}(M^\top \mathrm{diag}(q)^2 x) \\ &= T\ \mathrm{diag}(M^\top \mathrm{diag}(q)^2 \mathrm{diag}(x)\mathbf{1})\end{aligned}$$

$$
\begin{aligned}
&= T\ \mathrm{diag}(M^\top \mathrm{diag}(x)^2 \mathrm{diag}(q)^2 \mathbf{1}) \\
&= T\ \mathrm{diag}(M^\top \mathrm{diag}(x)(q_x^2)) \\
&= T\ \mathrm{diag}(M^\top x)\ q_x^2 \\
&= A_x\ q_x^2
\end{aligned}
$$

where we have used $a^\top \mathrm{diag}(x) b = (a^\top x)\ (b^\top x)$.

Our "starred" versions can be written in terms of the basic items T, M, U, π and q:

$$
\begin{aligned}
\mu^* &= U^\top \mathrm{diag}(q)^2 M \pi \\
\Sigma^* &= U^\top \mathrm{diag}(q)^2 M\ T\ \mathrm{diag}(\pi)\ M^\top \mathrm{diag}(q)^2 U \\
\Sigma^{*-1} &= (M^\top \mathrm{diag}(q)^2 U)^{-1}\ \mathrm{diag}(\pi)^{-1} \\
&\qquad \cdot\ T^{-1}\ (U^\top \mathrm{diag}(q)^2 M)^{-1} \\
K^*(x) &= U^\top \mathrm{diag}(q)^2 M\ T\ \mathrm{diag}(M^\top \mathrm{diag}(q)^2 U x) \\
&\qquad \cdot\ T\ \mathrm{diag}(\pi)\ M^\top \mathrm{diag}(q)^2 U
\end{aligned}
$$

So, we have

$$
c_1^* \equiv \mu^* = U^\top \mathrm{diag}(q)^2 M\ \pi
$$

likewise,

$$
\begin{aligned}
c_\infty^{*\top} &\equiv \mu^{*\top} \Sigma^{*-1} \\
&= \left(\pi^\top M^\top \mathrm{diag}(q)^2 U\right) \\
&\qquad \cdot\ \left((M^\top \mathrm{diag}(q)^2 U)^{-1}\ \mathrm{diag}(\pi)^{-1}\right. \\
&\qquad\qquad \left. \cdot\ T^{-1}\ (U^\top \mathrm{diag}(q)^2 M)^{-1}\right)
\end{aligned}
$$

$$
\begin{aligned}
&= \pi^\top \operatorname{diag}(\pi)^{-1}\, T^{-1}\, (U^\top \operatorname{diag}(q)^2 M)^{-1} \\
&= \mathbf{1}^\top T^{-1} (U^\top \operatorname{diag}(q)^2 M)^{-1} \\
&= \mathbf{1}^\top (U^\top \operatorname{diag}(q)^2 M)^{-1}
\end{aligned}
$$

For C^* we

$$
\begin{aligned}
C^*(y) &= K^*(y)\, \Sigma^{*-1} \\
&= U^\top \operatorname{diag}(q)^2 M\, T\, \operatorname{diag}(M^\top \operatorname{diag}(q)^2 U y) \\
&\qquad \cdot\, T\, \operatorname{diag}(\pi)\, M^\top \operatorname{diag}(q)^2 U\, \Sigma^{*-1} \\
&= U^\top \operatorname{diag}(q)^2 M\, T\, \operatorname{diag}(M^\top \operatorname{diag}(q)^2 U y) \\
&\qquad \cdot\, (U^\top \operatorname{diag}(q)^2 M)^{-1}
\end{aligned}
$$

Note that $U^* U^{*\top}$ is an $v \times v$ projection operator. Since its range is the same as that of M^* we have $M^{*\top} U^* U^{*\top} = M^{*\top}$. So, if $y^* = U^{*\top} \operatorname{diag}(q) x$, then:

$$
\begin{aligned}
C^*(y^*) &= U^\top \operatorname{diag}(q)^2 M\, T \\
&\qquad \cdot \operatorname{diag}(M^{*\top} U^* U^{*\top} \operatorname{diag}(q) x) \\
&\qquad\qquad \cdot\, (U^\top \operatorname{diag}(q)^2 M)^{-1} \\
&= U^\top \operatorname{diag}(q)^2 M\, T\, \operatorname{diag}(M^\top \operatorname{diag}(q)^2 x) \\
&\qquad \cdot\, (U^\top \operatorname{diag}(q)^2 M)^{-1} \\
&= (U^\top \operatorname{diag}(q)^2 M)\, A^*(x)\, (U^\top \operatorname{diag}(q)^2 M)^{-1}
\end{aligned}
$$

Hence equation (A.3) follows by a telescoping product.

□

Theorem 4. *Let X_t be generated by an $k \geq 2$ state HMM. Suppose we are given a U which has the property that range$(M) \subset$ range(U) and $|U_{ij}| \leq 1$. Suppose we use*

equation (A.3) to estimate $\lambda(x_1, x_2, \ldots, x_t)$ *based on* N *independent triples and for appropriate choice of* U^**. Then the following two inequalities*

$$\Lambda^* \geq \frac{6k}{\sigma_k^{*2}(\sqrt[2T+3]{1+\epsilon}-1)}\sqrt{\frac{2\log\frac{2k}{\delta}}{N}} \tag{A.4}$$

$$\sigma_k^* \geq 8k\sqrt{\frac{2\log\frac{2k}{\delta}}{N}}. \tag{A.5}$$

(where σ_k^* *is the smallest eigenvalue of* Σ^**) imply*

$$1-\epsilon \leq \left|\frac{\widehat{\lambda}(x_1,\ldots,x_t)}{\lambda(x_1,\ldots,x_t)}\right| \leq 1+\epsilon$$

or equivalently

$$1-\epsilon \leq \left|\frac{\widehat{\Pr}(x_1,\ldots,x_t)}{\Pr(x_1,\ldots,x_t)}\right| \leq 1+\epsilon$$

holds with probability at least $1-\delta$*.*

Proof:

The proof of this goes is identical to that given for theorem 3. The only worry is that we have defined y^*'s differently. But since we only required $|y| \leq 1$, and we have constructed $|y^*| \leq 1$, the Hoeffding inequality with elements of U still hold for U^*.

□

Details of generating the graphs

In lemma 10 and theorem 4 we see that we can increase our chances of obtaining a large enough Λ by multiplying each row of U by some function of that row. As long as we ensure that the elements of our new U^* are less than one, then we can make a claim on the accuracy of the relative "likelihood", and hence the relative probability,

generated by our sample.

Our figures utilize this gain in the size of Λ. For our corpus we use the Internet as captured by the Google n-gram dataset. We first create a dictionary of the $v-1$ most popular tokens, as well as an "out of vocabulary" token, for a final dictionary of size v. We take U to be the U matrix generated by the "thin" SVD of the P_{21} matrix generated using this vocabulary and Google 2-grams.

From this U we consider the first k columns. As per above, we can increase our chances of obtaining a large enough Λ by maximizing the size of the entries in this new $v \times k$ dimensional U matrix, hence we multiply each row by $1/\max_j(|U_{i,j}|)$, ensuring that at least one of the elements in our matrix is exactly 1 or -1. Now, using this new matrix U^* we use the frequencies from Google 1-grams, 2-grams, and 3-grams to compute μ^*, Σ^*, and K^* respectively, where each of the v vocabulary words (including one out-of-vocabulary token) correspond to a row of U^*. From this, we take Σ^{*-1} and compute the minimum element across μ^*, Σ^{*-1} and K^*.

We obtain σ_k^* in a similar way, first computing Σ^* from the appropriate $v \times k$ dimensional U^* matrix, then taking the SVD, recording the smallest singular value.

A.4 Supplement for Chapter 3

This appendix offers a sketch of the proof of Theorem 3. The proof uses the following definitions, which are slightly modified from those of Kakade and Foster (2007).

Definition 3. *Define* Λ *as the smallest element of* μ, Σ^{-1}, Ω^{-1}, *and* $K()$. *In other words,*

$$\Lambda \equiv \min\{\min_i |\mu_i|, \min_{i,j} |\Sigma^{-1}_{ij}|, \min_{i,j} |\Omega^{-1}_{ij}|,$$

$$\min_{i,j,k} |K_{ijk}|, \min_{i,j} |\Sigma_{ij}|, \min_{i,j} |\Omega_{ij}|, \}$$

where $K_{ijk} = K(\delta_j)_{ik}$ *are the elements of the tensor* $K()$.

Definition 4. *Define* σ_k *as the smallest singular value of* Σ *and* Ω.

The proof relies on the fact that a row vector multiplied by a series of matrices, and finally multiplied by a column vector amounts to a sum over all possible products of individual entries in the vectors and matrices. With this in mind, if we bound the largest relative error of any particular entry in the matrix by, say, ω, and there are, say, s parameters (vectors and matrices) being multiplied together, then by simple algebra the total relative error of the sum over the products is bounded by ω^s.

The proof then follows from two basic steps. First, one must bound the maximal relative error, ω for any particular entry in the parameters, which can be done using central limit-type theorems and the quantity Λ described above. Then, to calculate the exponent s one simply counts the number of parameters multiplied together when calculating the probability of a particular sequence of observations.

Since each hidden node is associated with exactly one observed node, it follows that $s = 12m + 2L$, where L is the number of levels (for instance in our example "Kilroy was here" there are two levels). s can be easily computed for arbitrary tree topologies.

It follows from chapter 2 that we achieve a sample complexity

$$N \geq \frac{128k^2s^2}{\epsilon^2\ \Lambda^2\sigma_k^4}\log\left(\frac{2k}{\delta}\right) \cdot \overbrace{\frac{\epsilon^2/s^2}{(\sqrt[s]{1+\epsilon}-1)^2}}^{\approx 1} \tag{A.6}$$

leading to the theorem stated above.

Lastly, note that in reality one does not see Λ and σ_k but instead estimates of these quantities; chapter 2 shows how to incorporate the accuracy of the estimates into the sample complexity.

Bibliography

Baum, L., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171.

Baum, L. E. and Eagon, J. A. (1967). An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bull. Amer. Math. Soc.*

Boots, B. and Gordon, G. (2011). An online spectral learning algorithm for partially observable nonlinear dynamical systems. *AAAI*.

Boots, B., Siddiqi, S., Gordon, G., and Smola, A. (2010). Hilbert space embeddings of hidden Markov models. *Proc. 27th Intl. Conf. on Machine Learning (ICML)*.

Brown, P., deSouza, P., Mercer, R., Pietra, V. D., and Lai, J. (1992). Class-based n-gram models of natural language. *Comput. Linguist.*, 18:467–479.

Carlyle, J. and Paz, A. (1971). Realizations by stochastic finite automata. *Journal of Computer and System Sciences*, 5(1):26–40.

Cohen, S., Stratos, K., Collins, M., Foster, D., and Ungar, L. (2012). Spectral learning of latent-variable PCFGs. In *Association of Computational Linguistics (ACL)*, volume 50.

Cohen, S. B., Stratos, K., Collins, M., Foster, D. P., and Ungar, L. (2013). Experiments with spectral learning of latent-variable PCFGs. *NAACL*.

Collins, M. (2000). Discriminative reranking for natural language parsing. In *ICML*, pages 175–182.

Collins, M. and Koo, T. (2005). Discriminative reranking for natural language parsing. *Comput. Linguist.*, 31(1):25–70.

Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., and Singer, Y. (2006). Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38.

Dhillon, P. S., Foster, D., and Ungar, L. (2011). Multi-view learning of word embeddings via CCA. In *Advances in Neural Information Processing Systems (NIPS)*, volume 24.

Eisner, J. (2000). Bilexical grammars and their cubic-time parsing algorithms.

Fliess, M. (1974). Matrices de hankel. *J. Math. Pures Appl*, 53(197-222):423.

Foster, D., Rodu, J., and Ungar, L. (2012). Spectral dimensionality reduction for HMMs. *ArXiV*.

Geman, S. and Geman, D. (1984). Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-6(6):721 –741.

Himmelmann, S. S. D. L. (2010). *HMM: Hidden Markov Models*.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):pp. 13–30.

Hsu, D., Kakade, S. M., and Zhang, T. (2009). A spectral algorithm for learning hidden Markov models. *COLT*.

Huang, F. and Yates, A. (2010). Open-domain semantic role labeling by modeling word spans. In *Association of Computational Linguistics (ACL)*.

Huang, X. D., Ariki, Y., and Jack, M. A. (1990). *Hidden Markov models for speech recognition*, volume 2004. Edinburgh university press Edinburgh.

Jaeger, H. (2000). Observable operator models for discrete stochastic time series. *Neural Computation*, 12(6).

Kakade, S. M. and Foster, D. P. (2007). Multi-view regression via canonical correlation analysis.

Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In *In Proc. ACL/HLT*.

Koo, T. and Collins, M. (2005). Hidden-variable models for discriminative reranking. In *HLT/EMNLP*.

Li, D., Miller, T., and Schuler, W. (2011). A pronoun anaphora resolution system based on factorial hidden Markov models. In *Association of Computational Linguistics (ACL)*.

Littman, M., Sutton, R., and Singh, S. (2002). Predictive representations of state. *Advances in neural information processing systems*, 2:1555–1562.

Luque, F., Quattoni, A., Balle, B., and Carreras, X. (2012). Spectral learning for non-deterministic dependency parsing. In *EACL*.

Marcus, M. P., Marcinkiewicz, M. A., and Santorini, B. (1993). Building a large annotated corpus of english: the penn treebank. *Comput. Linguist.*, 19:313–330.

Maskey, S. and Hirschberg, J. (2006). Summarizing speech without text using hidden Markov models. In *Association of Computational Linguistics (ACL)*.

McDonald, R. (2006). *Discriminative learning and spanning tree algorithms for dependency parsing*. PhD thesis, University of Pennsylvania, Philadelphia, PA, USA. AAI3225503.

Merialdo, B. (1994). Tagging english text with a probabilistic model. *Comput. Linguist.*, 20:155–171.

Musillo, G. A. and Merlo, P. (2008). Unlexicalised hidden variable models of split dependency grammars. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 213–216, Stroudsburg, PA, USA. Association for Computational Linguistics.

Parikh, A. P., Song, L., and Xing, E. P. (2011). A spectral algorithm for latent tree graphical models. In *ICML*, pages 1065–1072.

Petrov, S., Barrett, L., Thibaux, R., and Klein, D. (2006). Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 433–440, Stroudsburg, PA, USA. Association for Computational Linguistics.

Quattoni, A., Collins, M., and Darrell, T. (2004). Conditional random fields for object recognition. In *In NIPS*, pages 1097–1104. MIT Press.

Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286.

Ratnaparkhi, A. (1996). A Maximum Entropy Model for Part-Of-Speech Tagging. In Brill, E. and Church, K., editors, *Proceedings of the Empirical Methods in Natural Language Processing*, pages 133–142.

Robert Parker, e. a. (2009). English gigaword fourth edition. *Linguistic Data Consortium, Philadelphia*.

Rodu, J., Dhillon, P. S., Collins, M., Foster, D. P., and Ungar, L. H. (2012). Spectral dependency parsing with latent variables. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL'12.

Rodu, J., Foster, D. P., Wu, W., and Ungar, L. H. (2013). Using regression for spectral estimation of hmms. In *SLSP*, pages 212–223.

Rosenfeld, R. (1996). A maximum entropy approach to adaptive statistical language modeling. *Computer Speech and Language, 10:187228*.

Sangati, F., Zuidema, W., and Bod, R. (2009). A generative re-ranking model for dependency parsing. In *Proceedings of the 11th International Conference on Parsing Technologies*, IWPT '09, pages 238–241, Stroudsburg, PA, USA. Association for Computational Linguistics.

Schutzenbeegeb, M. (1961). On the definition of a family of automata. *Information and control*, 4(2-3).

Siddiqi, S., Boots, B., and Gordon, G. (2010). Reduced-rank hidden Markov models. *Proc. 13th Intl. Conf. on Artificial Intelligence and Statistics (AISTATS)*.

Song, L., Boots, B., Siddiqi, S., Gordon, G., and Smola, A. (2010). Hilbert space embeddings of hidden Markov models. *Proc. 27th Intl. Conf. on Machine Learning (ICML)*.

Terwijn, S. (2002a). On the learnability of hidden Markov models. *Grammatical Inference: Algorithms and Applications*, pages 344–348.

Terwijn, S. (2002b). On the learnability of hidden Markov models. In *Proceedings of the 6th International Colloquium on Grammatical Inference: Algorithms and Applications*, ICGI '02, pages 261–268, London, UK, UK. Springer-Verlag.

Thrun, S., Burgard, W., and Fox, D. (1998). A probabilistic approach to concurrent mapping and localization for mobile robots. *Autonomous Robots*, 5(3-4):253–271.

Turdakov, D. and Lizorkin, D. (2009). Hmm expanded to multiple interleaved chains as a model for word sense disambiguation. In *PACLIC*.

Zabokrtsky, Z. and Popel, M. (2009). Hidden Markov tree model in dependency-based machine translation. *ACL-IJCNLP*.

Zhao, S. and Gildea, D. (2010). A fast fertility hidden Markov model forword alignment using mcmc. *EMNLP*.