A Spectral Algorithm for Inference in Hidden semi-Markov Models

Igor Melnyk Arindam Banerjee
Department of Computer Science and Engineering, University of Minnesota, Minneapolis, USA
{melnyk, banerjee}@cs.umn.edu

Abstract

Hidden semi-Markov models (HSMMs) are latent variable models which allow latent state persistence and can be viewed as a generalization of the popular hidden Markov models (HMMs). In this paper, we introduce a novel spectral algorithm to perform inference in HSMMs. Our approach is based on estimating certain sample moments, whose order depends only logarithmically on the maximum length of the hidden state persistence. Moreover, the algorithm requires only a few spectral decompositions and is therefore computationally efficient. Empirical evaluations on synthetic and real data demonstrate the promise of the algorithm.

1 Introduction

Hidden semi-Markov models (HSMMs) are discrete latent variable models, which allow temporal persistence of latent states and can be viewed as a generalization of the popular hidden Markov models (HMMs) [6, 15, 22]. In HSMMs, the stochastic model for the unobservable process is defined by a semi-Markov chain: latent state at the next time step is determined by the current latent state as well as time elapsed since the entry into the current state. Ability to flexibly model such latent state persistence turns out to be useful in many application areas, including anomaly detection [19, 21], activity recognition [20], and speech synthesis [24].

Given a set of training sequences, one can formulate two distinct but related problems: learning, i.e., estimating model parameters and inference, i.e., computing the probability of an observed and/or latent variable sequence. The methods proposed for learning HSMM usually follow the initial idea due to Rabiner [18] based on the modifications of the Baum-Welch algorithm [5], which are all variants of the expectation maximization (EM) framework, presented in [7]. Once the parameters are estimated, we can then perform inference using, e.g., the forward-backward algorithm of Yu et al. [23]. However, since EM, in general, has no guarantees in estimating the parameters correctly and can suffer from slow convergence, such methods can be inefficient and/or inconsistent.

In recent years, there has been an increased interest in spectral algorithms, which provide computationally efficient, local-minimum-free, provably consistent algorithms for parameter estimation and/or inference. For example, Anandkumar et al. [2, 3] have proposed spectral methods for learning the parameters of a wide class of tree-structured latent graphical models, including Gaussian mixture models, topic models, and latent Dirichlet allocation. Hsu et al. [8] have proposed an efficient spectral algorithm for inference in HMMs. The algorithm learns a so called observable representation and uses it to do inference on observable variables. The approach, however, was specific to HMMs and not easily extendable to other latent variable graphical models. Parikh et al. [17] have introduced a spectral algorithm to perform inference in latent tree graphical models with arbitrary topology, and later in [16] a general spectral inference framework for latent junction trees.

In this paper, we utilize the framework of [16] and introduce a novel spectral algorithm for inference in HSMMs. Since we address a more specific problem than [16], our results shed more light into the details of the spectral framework for HSMMs, allow for a sharper analysis, and yield a significantly more efficient algorithm than the general framework in [16]. There are two main technical contributions in this work. First, by exploiting the homogeneity of HSMMs we make our algorithm more efficient and accurate than if we directly follow the recipe in [16] for general graphs. In
A Spectral Algorithm for Inference in Hidden semi-Markov Models

particular, our approach ensures that the number of matrix multiplications and inverses is fixed and independent of sequence length. Second, we show that the order of tensors in estimated observable representation depends only logarithmically on the maximum length of latent state persistence. In experiments, comparing our method with EM on both synthetic and real datasets, two observations stand out: first, the spectral method gets similar or better performance than EM as the number of samples increases, and the spectral method is orders of magnitude faster than EM for the datasets we consider.

Few remarks are in order about the proposed algorithm. Note that our method does not estimate model parameters explicitly but rather learns alternative representation to perform inference on observable variables. Moreover, our formulation cannot be directly used to infer hidden states, although methods such as in [14] can be potentially utilized to recover original HSMM parameters from the learned representation.

The rest of the paper is organized as follows: We introduce notation in Section 2. The inference problem and the proposed algorithm are presented in Sections 3 and 4. In Section 5, we discuss the analysis of the algorithm, followed by evaluations in Section 6 and conclusion in Section 7. Most of the technical analysis, proofs, and additional details can be found in [12].

2 Notation and Preliminaries

In this section, we cover the basic facts about the tensor algebra; more details can be found in [12], while a detailed tutorial on tensors is in [9] or [10]. A tensor is defined as a multidimensional array of data, denoted by boldface script letters, e.g., $\mathbf{X}_{p_1,\ldots,p_K,q_1,\ldots,q_L} \in \mathbb{R}^{I_{p_1} \times \cdots \times I_{p_K} \times I_{q_1} \cdots I_{q_L}}$, which is $N$-th order tensor of dimensions $I_{m_1} \times \cdots \times I_{m_N}$. A specific dimension (or mode) is denoted by the subscript variable $m_i$, whose size is $I_{m_i}$.

Any tensor can be matrisized (or flattened) into a matrix. If we split the modes into two disjoint sets, one corresponding to rows and the other to columns, e.g., $\{m_1,\ldots,m_N\} = \{p_1,\ldots,p_K\} \cup \{q_1,\ldots,q_L\}$, then a matrisization of $\mathbf{X}$ is $\mathbf{X}_{p_1,\ldots,p_K,q_1,\ldots,q_L} \in \mathbb{R}^{I_{p_1} \cdots I_{p_K} \times I_{q_1} \cdots I_{q_L}}$. Multiplication of tensors is performed along specific modes. For this, we flatten the tensor to a matrix with appropriate choice for rows and columns, perform matrix multiplication and transform the result back to tensor. The multiplication is denoted by a symbol $\times$ with an optional subscript representing the modes along which the operation is performed, e.g., multiplication along $q_1,\ldots,q_L$:

$$\mathbf{Z}_{p_1,\ldots,p_K,r_1,\ldots,r_M} = \mathbf{X}_{p_1,\ldots,p_K,q_1,\ldots,q_L} \times_{q_1,\ldots,q_L} \mathbf{Y}_{q_1,\ldots,q_L,r_1,\ldots,r_M},$$

where $\mathbf{Y} \in \mathbb{R}^{I_{q_1} \times \cdots \times I_{q_L} \times I_{r_1} \times \cdots \times I_{r_M}}$ and the resulting tensor is $\mathbf{Z} \in \mathbb{R}^{I_{p_1} \times \cdots \times I_{p_K} \times I_{r_1} \times \cdots \times I_{r_M}}$.

An important fact about tensor multiplication is that in a series of tensor multiplications the order is irrelevant as long as the multiplication is done along the matching modes: $\mathbf{X}_{sp} \times_q \mathbf{Y}_{tr} \mathbf{Z}_{rs} = (\mathbf{X}_{sp} \times_q \mathbf{Z}_{rs}) \times_r \mathbf{Y}_{tr}$.

Finally, we discuss the operation of tensor inversion. Tensor inverse $\mathbf{X}^{-1}$ is always defined with respect to a certain subset of the modes:

$$\mathbf{X}_{p_1,\ldots,p_K,q_1,\ldots,q_L} \times_{q_1,\ldots,q_L} \mathbf{X}^{-1}_{p_1,\ldots,p_K,q_1,\ldots,q_L} = \mathbf{I}_{p_1,\ldots,p_K},$$

where the inversion is performed along the modes $q_1,\ldots,q_L$. Tensor on the right hand side can also be written as $\mathbf{I}_{p_1,\ldots,p_K}$ by dropping the duplicated modes.

To perform tensor inversion, we first matrisize it. If the modes to be inverted are associated with columns of the matrix, we compute the right matrix inverse, so that these modes get eliminated after the product. Otherwise, if those modes associated with rows, we compute left matrix inverse. For example, in the above equation the matrisized tensor might be of the form $\mathbf{X} \in \mathbb{R}^{I_{p_1} \cdots I_{p_K} \times I_{q_1} \cdots I_{q_L}}$, and we would compute the right matrix inverse so that the modes $q_1,\ldots,q_L$ are eliminated. If $\mathbf{X}$ has full row rank, then we compute its inverse, otherwise the pseudo-inverse.

Tensorizing matrix $\mathbf{X}^{-1}$ gives us desired tensor inverse.

3 Problem Formulation: Inference in HSMMs

In this paper, we consider the problem of inference in HSMM. From a graphical model perspective, HSMM has three sets of variables: the observations $o_t \in \{1,\ldots,n_o\}$, the latent states $x_t \in \{1,\ldots,n_x\}$, and another latent variable $d_t \in \{1,\ldots,n_d\}$ which determines the length of state persistence. HSMM is specified by three conditional probability tables (CPTs): the observation/emission probability $p(o_t | x_t)$ and the state transition and the duration probabilities given by:

$$p(d_t | x_t, d_{t-1}) = \begin{cases} p(d_t | x_t) & \text{if } d_{t-1} = 1 \\ \delta(d_t, d_{t-1}-1) & \text{if } d_{t-1} > 1 \end{cases}$$

$$p(x_t | x_{t-1}, d_{t-1}) = \begin{cases} p(x_t | x_{t-1}) & \text{if } d_{t-1} = 1 \\ \delta(x_t, x_{t-1}) & \text{if } d_{t-1} > 1 \end{cases},$$

where $\delta(a, b)$ denotes the Dirac delta function: $\delta(a, b) = 1$ if $a = b$ and 0 otherwise. In addition, one can consider suitable prior probabilities $p(x_0)$ and $p(d_0)$. In essence, $d_t$ works as a down counter for state persistence. When $d_{t-1} > 1$, the model remains in the same state $x_t = x_{t-1}$, while when $d_{t-1} = 1$, one samples a new state $x_t$ and the new duration in that state.
For our analysis, we assume \( p(d_t|x_t, d_{t−1}) = 1 \) to be a multinomial distribution over \( \{1, \ldots, n_d\} \) where \( n_d \) denotes the largest duration of state persistence.

The considered inference problem can be posed as follows: given a set of sequences \( \{S^1, \ldots, S^N\} \) drawn independently from the HSM model, where each sequence is \( S^i = (o^i_1, \ldots, o^i_T), i = 1, \ldots, N \), our goal is to develop a provably correct spectral algorithm for computing \( p(S^{test}) \) of any given test sequence \( S^{test} = (o^{test}_1, \ldots, o^{test}_T) \).

We start by considering the matrix forms of the HSM parameters and writing the computations in tensor notation, as introduced in Section 2. Specifically, \( p(d_t|x_t, d_{t−1} = 1) \) is denoted as \( D \in \mathbb{R}^{n_d \times n_d} \), \( p(x_t|x_{t−1}, d_{t−1} = 1) \) is denoted as \( X \in \mathbb{R}^{n_x \times n_x} \), and \( p(o_t|x_t) \) as \( O \in \mathbb{R}^{n_o \times n_x} \). We make the following assumptions on the HSM parameters:

**Assumptions 3.1**

1. \( X \) is full rank and has non-zero probability of visiting any state from any other state.
2. \( D \) has a non-zero probability of any duration in any state.
3. \( O \) is full column rank and \( n_x \leq n_o \).

To express the joint probability \( p(o_1, \ldots, o_T) \) for an observation sequence in tensor form, we utilize the junction tree [4] corresponding to the graphical model of HSM (see Figure 1). We proceed by embedding the clique CPTs of the junction tree into tensors. For example, the clique \( X_t \), containing the CPT of \( p(x_t|x_{t−1}, d_{t−1}) \) is embedded into tensor \( X_{t−1} \).

For ease of exposition, the tensor’s modes are named based on the variables on which the tensor depends. Similarly, we embed the clique \( D_t \) with its CPT \( p(d_t|x_t, d_{t−1}) \) into tensor \( D_{t−1} \), and \( O_t \) containing \( p(o_t|x_t) \) into tensor \( O_{t−1} \).

If we denote the joint probability of the observed sequence \( p(o_1, \ldots, o_T) \) as \( P \) then the message passing algorithm for the junction tree in Figure 1 can be represented as tensor multiplications:

\[
P_{o_1,\ldots,o_T} = \prod_t p(d_t|x_t, d_{t−1}) D_t p(x_t|x_{t−1}, d_{t−1}) X_t p(o_t|x_t) O_t \]

where, for simplicity, we denoted by \( \prod_t \), the tensor product over multiple time steps. Observe that the tensors are multiplied along the modes (dimensions) which are the separator variables between the cliques in Figure 1. A certain mode of the tensor is duplicated the number of times such variable appears in the separators adjacent to the clique, ensuring that tensor multiplication remains valid. In what follows, we represent expression (3) in the observable form so that all the factors can be estimated directly from data using certain sample moments and provide a practical algorithm implementing these ideas.

### 4 Spectral Algorithm for Inference in HSMM

Observe that the expression for the joint probability in (3) depends on the unknown model parameters. Our goal is to change the tensor representation such that \( P \) can be written in terms of the quantities directly computable from data. To that end, we follow [16] and between every two neighboring factors in (3) introduce an identity tensor with the modes corresponding to the modes along which the multiplication is performed. Intuitively, this operation corresponds to the marginalization step, expressed in tensor form. For example, consider a part of expression (3) after introducing identity tensors:

\[
\times x_{t−1} d_{t−1} X_{t−1} X_{t−1} D_{t−1} D_{t−1} O_{t−1} O_{t−1}
\]

where all the identity tensors have duplicated modes but are not shown. Now rewrite each of the identity tensors as a multiplication of some factor times its inverse. For example, \( F_1 = F_1^{-1} \times w_{x_1} x_{t−1} x_{t−1} \). Note that the choice of mode \( x_t \)
is fixed and is determined by the modes of the identity tensor $\mathcal{I}$, while the mode $\omega_{x_i}$ is not fixed and we have a freedom in selecting it. Moreover, since the tensor inversion is done along the mode $\omega_{x_i}$, if the matrix $\mathbf{F}$ has its rows associated with $\omega_{x_i}$, then this matrix must have full column rank for the inverse to exist and for the product $\mathbf{F}^{-1} \mathbf{F}$ to equal identity matrix (see Section 2). Based on this, we choose the modes $\omega_{x_i}$ such that (i) $\omega_{x_i}$ are the observed variables, (ii) $\mathbf{F}$ is invertible, and (iii) interpret this factor as corresponding to a conditional probability distribution, i.e., $p(\omega_{x_i}|x_i)$ and write as $\mathbf{F}_{\omega_{x_i}|x_i}$.

After expanding each of the identity tensors, regrouping the factors and recalling that in a series of tensor multiplication the order is irrelevant, we can identify the tensors

$$
\mathbf{D} = \mathbf{F}^{-1} \mathbf{D} \mathbf{F} = \mathbf{F}^{-1} \mathbf{F}_{\omega_{x_i}|x_i} \times x_{t-1} | d_{t-1}, x_{t-1} | d_{t-2}, x_{t-1} | d_{t-1}, x_{t-1} | d_{t-2},
$$

$$
\mathbf{X} = \mathbf{F}^{-1} \mathbf{X} \mathbf{F} = \mathbf{F}^{-1} \mathbf{F}_{\omega_{x_i}|x_i} \times x_{t-1} | d_{t-1}, x_{t-1} | d_{t-2},
$$

$$
\mathbf{O} = \mathbf{F}^{-1} \mathbf{O} \mathbf{F} = \mathbf{F}^{-1} \mathbf{F}_{\omega_{x_i}|x_i} \times x_{t-1} | d_{t-1}, x_{t-1} | d_{t-2},
$$

and $\mathbf{O}_{\omega_{x_i}|x_i} = \mathbf{F}^{-1} \mathbf{O} \mathbf{F}_{\omega_{x_i}|x_i}$. Note that although each of the above tensors depends only on the observed variables $\omega$, it is not clear yet how to estimate them: the expressions on the right depend on the unknown model parameters, while the tensors on the left do not correspond to valid probability distributions (due to the presence of inverses $\mathbf{F}^{-1}$).

For example, $\omega_{x_i|d_{t-2}}$ is not a tensor form of $p(\omega_{x_i|d_{t-2}}, \omega_{x_{t-1}|d_{t-1}})$. Next, we discuss the choice of the observable set $\omega$ in the factors $\mathbf{F}$. From Figure 1 we can see that there are three types of separators: $x_{t-1} d_{t-1}$, $x_t d_{t-1}$ and $x_t$, consequently, there are three types of identity tensors which we introduced: $\mathcal{I}$, $\mathcal{J}$ and $\mathcal{K}$.

Therefore, we need to define three types of observable sets $\omega_{x_{t-1} d_{t-1}}$, $\omega_{x_t d_{t-1}}$ and $\omega_{x_t}$. There could be multiple choices for these sets, one of them is $\omega_{x_{t-1} d_{t-1}} = \{o_{t-1}, o_{t-2}, \ldots\}$ and $\omega_{x_t} = o_t$ for all $t$. The detailed description of how and what number of these observations to select is deferred until Section 5. In what follows, we define $\mathbf{O}_{\mathcal{R}_t} := \{o_{t+1}, o_{t+2}, \ldots\}$, to emphasize that this is a set of observations starting at time stamp $t+1$ and going to the right (or forward in time), see Figure 2. With these definitions, we can now rewrite (3) in the form:

$$
\mathcal{P} = \prod_{t=1}^T \mathbf{O}_{\mathcal{R}_t} \mathbf{O}_{\mathcal{R}_t-1} \mathbf{O}_{\mathcal{R}_t} \mathbf{X}_{\omega_{x_t}|x_t} \mathbf{X}_{\omega_{x_t}|x_t} \mathbf{F}_{\omega_{x_t}|x_t} \times x_{t-1} | d_{t-1}, x_{t-2} | d_{t-2}, x_{t-1} | d_{t-2},
$$

(4)

Comparing (3) and (4) we see that the above equation expresses the joint probability distribution in the observable form. As noted above, we cannot yet use this formula in practice since we do not know how to compute the transformed tensors. In what follows, we show how to estimate such tensors directly from data, without the need of the model parameters.

4.1 Estimation of Observable Tensors

Consider the tensor $\mathbf{D}$:

$$
\mathbf{D}_{\mathcal{R}_t} = \mathbf{D}_{\mathcal{R}_t} \mathbf{F}_{\omega_{x_t}|x_t} \times x_{t-1} | d_{t-1}, x_{t-2} | d_{t-2}, x_{t-1} | d_{t-2},
$$

(5)

whose modes are the observable variables $\mathbf{O}_{\mathcal{R}_t-1}$ and $\mathbf{O}_{\mathcal{R}_t}$. To estimate this tensor from data, consider $\mathbf{O}_{\mathcal{R}_t-1}$, a set of the observed variables such that $\mathbf{O}_{\mathcal{R}_t-1}$ and $\mathbf{O}_{\mathcal{R}_t}$ are independent, conditioned on $x_{t-1} d_{t-2}$ (see Figure 2). The tensor form of this relationship is:

$$
\mathbf{M}_{\mathcal{R}_t} = \mathbf{F}_{\omega_{x_t}|x_t} \times x_{t-1} | d_{t-1}, x_{t-2} | d_{t-2}, x_{t-1} | d_{t-2},
$$

(6)

where tensor $\mathbf{K}$ represents the marginal $p(x_{t-1}, d_{t-2})$. Note that, though not shown, the modes $x_{t-1}$ and $d_{t-2}$ need to appear twice in $\mathbf{K}$, since it interacts with both other terms. The set $\mathbf{O}_{\mathcal{R}_t-1}$ is defined in a way similar to $\mathbf{O}_{\mathcal{R}_t}$ but with the set of observations starting at time stamp $t-2$ and going to the left (or backward in time), i.e., $\mathbf{O}_{\mathcal{R}_t-1} := \{o_{t-3}, o_{t-2}\}$ (see Figure 2).

Next, express the inverse of the tensor $\mathbf{O}_{\mathcal{R}_t-1} \times x_{t-1} | d_{t-2}$.
from (6) and substitute back to (5).
\[
\tilde{D}_{t-1} = \frac{M^{-1}}{O_{t-1}} \times x_{t-1},
\]
where we eliminated all the latent variables by multiplying the last four terms on the first line. Observe that the tensors \(M_{t-1} \) and \(M_t \) represent valid probability distributions and though they are defined using unknown model parameters, we can readily estimate them from data. For example, \(O_{t-1} \) is a tensor, where each entry is computed from the frequency of co-occurrence of tuples of the observed symbols \(\{\ldots, o_{t-3}, o_{t-2}, o_{t+1}, o_{t+2}, \ldots\} \). The specific number and order of these symbols will be determined in Section 5. Similar derivations can be used to find the estimates for tensors \(\tilde{X} \) and \(\tilde{O} = M^{-1} \times x_{t-1} \), (see [12] for details).

4.2 Spectral Algorithm

In the previous section, we expressed the tensors \(D, X \) and \(O \) in terms of the moments directly computable from data, so now we can obtain the spectral algorithm to compute \(p \) entirely using the observed variables. The basic version of the spectral HSMM algorithm, which follows from the framework of [16], can be described as a two step process: in the learning step, compute \(D, X \) and \(O \) for all \(t \) using the data. In the inference step, use (4) to compute \(p(S_{text}) \).

If we denote the number of required observations in \(O \) as \(\ell \) (in Section 5 we will show that \(\ell = [1 + \frac{\text{log} n_{text}}{\text{log} 2}] \)), then the algorithm’s computational complexity is \(O \left(\left(n_{text}^{3\ell} + N\ell\right)T\right) \) for learning and \(O \left(n_{text}^{3\ell}T\right) \) for inference, mainly determined by tensor inversions, multiplications, and the estimation of tensors \(M \) in (7) and in \(X \) and \(O \) for all \(t \). Here, \(N \) is the number of training samples and \(T \) is the length of the observation sequences. Note that for large \(\ell \) accurate estimation of tensors \(M \) for each \(t \) will require large number of training sequences which might not be available, leading to inaccurate and unstable computations.

A novel aspect of our work is the improvement of the accuracy and efficiency of the basic algorithm mentioned above by exploiting the homogeneity property of HSMM and estimating the tensors \(X, D \) and \(O \) in the batch, by averaging across all \(t \). Thus, we compute only three tensors for all \(t \), as opposed to computing these tensors for each \(t \). For example, using (7), the batch form of tensor \(D \) takes a form:

\[
D = \left(\frac{1}{T} \sum_t o_{t-1} M_{t-1} o_{rt} \right)^{-1} \times \left(\frac{1}{T} \sum_t o_{t-1} M_{t} \right),
\]

where \(O_L \) denotes a generic mode of the averaged tensor \(M \), corresponding to \(O_{rt} \) for all \(t \). Similar expressions can be defined for other tensors: the detailed derivations are provided in [12]. This modification reduced the computational complexity of the learning phase to \(O \left(n_{text}^{3\ell} + N\ell\right)T \) (the cost of inference step still remains \(O \left(n_{text}^{3\ell}T\right) \), where the main operations are now tensor additions and estimation of tensors \(M \).

Note that the number of inverses and multiplications in the learning phase is now fixed and independent of sequence length. This is in contrast to the basic version of algorithm, which involves tensor inverses and multiplications at every step \(t \) (e.g., see (7)). Moreover, such averaging improves the accuracy of the resulting algorithm since the estimates obtained in this form have lower variance, which in turn ensures that the computed inverses are more stable and accurate.

5 Rank Analysis of Observable Tensors

In Section 4.1, when we derived \(D, X \) and \(O \), we glossed over the question of the existence of tensor inverses \(M^{-1} \). In this section, our task is to analyze the rank structure of these tensors and impose restrictions on the sets \(L \) and \(R \) to ensure that the rank conditions are satisfied. For example, consider equation (7) and expand all its terms using (6) to get

\[
D = M^{-1} \times \left(\frac{1}{T} \sum_t o_{rt} M_{rt} \right) \times \left(\frac{1}{T} \sum_t o_{t-1} M_{t-1} \right) \times \left(\frac{1}{T} \sum_t o_{t} M_{t} \right),
\]

where we dropped the multiplication subtensors and the duplicated modes, which can be inferred from the context. Observe, that in order for the above equation to produce (5), the terms in the middle must multiply out into identity tensor, i.e.,

\[
J_{x_{t-1}d_{t-2}} = K^{-1}_{x_{t-1}d_{t-2}} \times D_{x_{t-1}d_{t-2}} \times K^{-1}_{x_{t-1}d_{t-2}} \times J_{x_{t-1}d_{t-2}} \times \left(\frac{1}{T} \sum_t o_{t-1} M_{t-1} o_{rt} \right) \times \left(\frac{1}{T} \sum_t o_{t} M_{t} \right) \times \left(\frac{1}{T} \sum_t o_{t-1} M_{t-1} o_{rt} \right) \times \left(\frac{1}{T} \sum_t o_{t} M_{t} \right),
\]

Moreover, recall that \(J_{x_{t-1}d_{t-2}} \) was originally introduced as part of the identity tensor

\[
J_{x_{t-1}d_{t-2}} = o_{rt} o_{rt} o_{rt} \times \left(\frac{1}{T} \sum_t o_{t-1} M_{t-1} o_{rt} \right) \times \left(\frac{1}{T} \sum_t o_{t} M_{t} \right),
\]

694
therefore, we can conclude that for (7) to exist, the identity statements in (8), (9) and (10) must be satisfied. These statements have implications for the ranks of $K_{x_1\ldots d_2}$, $O_{L_{t\ldots d_2}}$, $F_{t\ldots d_2}$, and $F_{r\ldots d_2}$, which in turn determine the length of the observation sequences $O_{L_{t\ldots d_2}}$ and $O_{R_{t\ldots d_2}}$.

Since $K_{x_1\ldots d_2}$ represents a distribution $p(x_t\ldots d_2)$, its matrisized version is a diagonal matrix with $p(x_{t-1}\ldots d_2)$ on the diagonal. Using statements 1 and 2 in Assumptions 3.1, it can be concluded that the diagonal elements in this matrix are non-zero and it has rank $n_xn_d$. It is thus invertible and so (8) is satisfied.

Next, consider (9) and recall from Section 2 that if we select the observations $o_{t\ldots d_2}$ for the corresponding tensors to have rank $n_x$, then the sequence of hidden states $O_{R_{t\ldots d_2}}$ has rank $n_xn_d$ and its matrisized version is a diagonal matrix with $n_x$ at the $d_2$-th position. These statements have implications for the identity statements in (8), (9) and (10) must be satisfied.

5.1 Proof sketch of Theorem 5.1

In the following we present main ideas to prove Theorem 5.1, all the details can be found in [12]. Define by $X_{R_{t\ldots d_2}} = \{x_t, x_{t+1}, \ldots\}$, the sequence of hidden states corresponding to $O_{R_{t\ldots d_2}} = \{o_t, o_{t+1}, \ldots\}$. Using conditional independence property of graphical model in Figure 1, i.e., $O_{R_{t\ldots d_2}}$ and $x_{t-1}\ldots d_2$ are independent given $X_{R_{t\ldots d_2}}$, we can write:

$$O_{R_{t\ldots d_2}} = F_{R_{t\ldots d_2}} = O_{R_{t\ldots d_2}}O_{R_{t\ldots d_2}}^T,$$

for some tensors $O_{R_{t\ldots d_2}}$ and $F_{R_{t\ldots d_2}}$, representing the corresponding probability distributions.

Denoting $\ell = |O_{R_{t\ldots d_2}}| = |X_{R_{t\ldots d_2}}|$, it can be verified that the matrisized form of $O_{R_{t\ldots d_2}}$ in (11) can be written as $Q = \otimes_0\otimes_{i=1}^{\ell}O_i \in \mathbb{R}^{n_x \times n_x}$, a Kronecker product of the observation matrix $O$ with itself $\ell$ times. According to the Assumptions 3.1, $\text{rank}(O) = n_x$ and $n_x \leq n_0$, and using the rank property of the Kronecker product, we infer that $\text{rank}(Q) = n_x^\ell$.

Combining the above conclusion with the fact that the matrisized form of the other two tensors in (11) is $F \in \mathbb{R}^{n_x \times n_x}$ and $T \in \mathbb{R}^{n_x \times n_x}$, to ensure the rank of $F$ is $n_x$, we need to select a set of variables $X_{R_{t\ldots d_2}}$ so that $\text{rank}(X_{R_{t\ldots d_2}}T) = n_x$ with the condition that $n_x^\ell \geq n_xn_d$. Thus, the problem of the analysis of the rank structure of tensor $O_{R_{t\ldots d_2}}$ was translated to the problem of rank structure of matrix $X_{R_{t\ldots d_2}}T$. The main idea of the analysis of such a matrix is to understand the mechanism how the rank changes as the size of $X_{R_{t\ldots d_2}}$ increases.

Starting with a matrix $T$, its rank is $n_x$ since it is a matrisized version of the transition probability of the model and statement 1 in Assumptions 3.1 guarantees it has full rank. Next, if we add a hidden state.
at the consecutive time stamp, i.e., $T_{x+1,x|t-1,t-2} \in \mathbb{R}^{n_x^2 \times n_x n_d}$ then it turns out that the rank of this matrix is $2n_x$ (assuming that $n_x n_d \geq 2n_x$). In general, it can be shown that the matrix $T$ constructed with the hidden states at consecutive time stamp allows only linear growth of its rank. Consequently, to have rank $n_x n_d$ would require $\ell = n_d$ observations.

On the other hand, if we consider $T_{x+\ell,x|t-1,t-2} \in \mathbb{R}^{n_x^2 \times n_x n_d}$, for $\delta = 1, \ldots, n_x - 1$, then the rank of this matrix is $(\delta + 1)n_x$, with the largest rank of $n_x^2$ (if $n_x n_d \geq n_x^2$). In general, it can be proved that if we include hidden state at time stamp $j$-th, $j = 1, \ldots, \ell$ by skipping $n_x^2 - n_x^2 - 1$ time stamps, we will be increasing rank as $n_x^2$, i.e., exponentially fast up to a maximum achievable rank $n_x n_d$. As a result, the number of required observations is only $\ell = \lceil 1 + \log n_d \log n_x \rceil$. To illustrate this, refer back to Figure 3 where $n_x = 3$ and $n_d = 20$. The $\ell = 4$ observations we include for $O_{t-1}$ are $a_{t+10}$, $a_{t+17}$, $a_{t+11}$ and $a_t$, which lead to the following rank growth: 3, 9, 27, 60.

6 Experiments

In this section we evaluated the performance of the proposed algorithm both on synthetic as well as real datasets and compared its performance to a standard EM algorithm.

6.1 Synthetic Data

Using synthetic data, we compared the estimation accuracy and the runtime of the spectral algorithm with EM. For this, we defined two HSMMs, one with $n_o = 3, n_x = 2, n_d = 2$ (top row) and $n_o = 5, n_x = 4, n_d = 6$ (bottom row). (a), (d): Error for EM across different iterations for various training datasets. The straight lines show the performance for spectral method. (b), (e): Average error and one standard deviation over 100 runs for EM after convergence and spectral algorithm across different number of training data. (c), (f): Runtime, in seconds, for both methods.

Figure 4: Performance of the spectral algorithm and EM on synthetic data generated from HSMM with $n_o = 3, n_x = 2, n_d = 2$ (top row) and $n_o = 5, n_x = 4, n_d = 6$ (bottom row). (a), (d): Error for EM across different iterations for various training datasets. The straight lines show the performance for spectral method. (b), (e): Average error and one standard deviation over 100 runs for EM after convergence and spectral algorithm across different number of training data. (c), (f): Runtime, in seconds, for both methods.

It can be observed from plots (b) and (e) in Figure 4 that with the small training set, EM achieves smaller errors, while as the number of training samples increases, the spectral method becomes more accurate, outperforming EM. Also, comparing the plots (a), (b) with (d) and (e), we can conclude that for larger models the spectral method requires more data in order to achieve same or better accuracy than EM. This is expected since the sizes of estimated tensors grow with the model size. Moreover, the plots (c) and (d) in Figure 4 show that spectral method is several orders of magnitude faster than EM. Given the above results, we can conclude that for small datasets EM is a preferable algorithm, while for large data, the spectral algorithm is a better choice, it achieves higher accuracy and requires significantly smaller computational resources.

6.2 Real Data

We also compared the performance of the spectral algorithm and EM on real NASA flight dataset [1], containing over 180000 flights of 35 aircrafts from a de-
A Spectral Algorithm for Inference in Hidden semi-Markov Models

Figure 5: Evaluation of the spectral algorithm and EM on real data. (a) and (b): Normalized joint loglikelihood computed by spectral algorithm (a) and EM (b) for a set of 200 test flights, with 100 normal and 100 anomalous. HSMM parameters: $n_o = 9, n_x = 8, n_d = 40$ (c): The Receiver Operating Characteristic (ROC) curve, illustrating classification accuracy of the algorithms.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>$n_o = 9, n_x = 8, n_d = 40$</th>
<th>$n_o = 9, n_x = 7, n_d = 30$</th>
<th>$n_o = 9, n_x = 6, n_d = 20$</th>
<th>$n_o = 9, n_x = 5, n_d = 10$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>Spectral 6.8 hours &gt; 2 days</td>
<td>EM 6.4 hours &gt; 2 days</td>
<td>EM 6.4 hours &gt; 2 days</td>
<td>EM 6.3 hours &gt; 2 days</td>
</tr>
<tr>
<td>AUC</td>
<td>Spectral 0.9066</td>
<td>EM 0.8010</td>
<td>EM 0.7215</td>
<td>EM 0.9019</td>
</tr>
</tbody>
</table>

Table 1: Comparison of AUC scores and running time for EM and spectral algorithm for various model parameters.

The data has a record of 186 parameters, sampled at 1 Hz, including sensor readings and pilot actions. We considered a problem of anomaly detection in aviation systems [11] and used HSMM to detect abnormal flights based on pilot actions. Specifically, we modeled the phases of the flight as hidden states and the pilot actions as the observations from these phases (see [13] for more details). We focused on a part of flight related to approach (15 – 60 minutes in duration) for a subset of flights landing at the same airport. We chose 9 pilot commands, among which are “selected altitude”, “selected heading”, etc.

A simple data filter, based on the histogram of the pilot actions, was applied to select 10020 normal flights for training. A test set contained 200 flights, with 100 of them being similar to the training set and the rest were selected from the flights rejected by the filter. Most of abnormal flights contained low occurrence events, such as fast descent, unusual usage of air brakes, etc., and few significant anomalies, e.g., the aborted descent in order to delay the flight. The length of the considered sequences varied anywhere from 500 to 4000 time stamps.

We applied EM and spectral algorithm to compute the normalized joint log-likelihood of the observed pilot actions, Figure 5 shows the results. The high-likelihood sequences were considered normal and low-likelihood ones classified as anomalous (see plots (a) and (b)). Both algorithms achieved similar detection accuracy, with the spectral algorithm having the Area Under Curve (AUC) score of 0.91 and the EM had AUC = 0.89. On the other hand, the computational time of the spectral algorithm was orders of magnitude smaller as compared to EM (see third column in Table 1). We also compared performance of both algorithm on the same flight data while varying the dimensionality of the HSMM parameters (see Table 1). We can see that although the performance of EM and spectral algorithm is similar across many models, the latter offers significant computational savings.

7 Conclusion

In this paper, we present a novel spectral algorithm to perform inference in HSMM. Our approach is based on estimating certain sample moments of size logarithmic in maximum state persistence and requires fixed number of matrix multiplications and inverses, independent of sequence length. Empirical evaluation on synthetic and real datasets illustrate the promise of the proposed spectral algorithm, especially for large datasets. Going forward, we plan to explore if similar spectral methods can be developed for inference in more general dynamic Bayesian networks.

Acknowledgements

We thank reviewers for helpful comments and suggestions. The research was supported by NSF grants IIS-1447566, IIS-1422557, CCF-1451986, CNS-1314560, IIS-0953274, IIS-1029711, and by NASA grant NNX12A3Q35A. Authors acknowledge technical support from the University of Minnesota Supercomputing Institute.
References


