A Spectral Algorithm for Inference in Hidden Semi-Markov Models
Igor Melnyk and Arindam Banerjee
Department of Computer Science and Engineering
University of Minnesota, Twin Cities
July 15, 2014

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. Unlike expectation maximization (EM), our approach correctly estimates the probability of given observation sequence based on a set of training sequences. 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.

Keywords: Graphical model, hidden semi-Markov model, spectral algorithm, tensor analysis

1 Introduction

Hidden semi-Markov models (HSMM) 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) \cite{rabiner1989}. In HSMMs, the stochastic model for the unobservable process is defined by a semi-Markov chain, so that the 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. The ability to flexibly model such latent state persistence turns out to be useful in many application areas, including anomaly detection \cite{yu2002}, activity recognition \cite{yu2003}, and speech synthesis \cite{chung2002}. Such state persistence is in contrast to HMMs, which use a Markov chain over latent state transitions and hence have an implicit geometric distribution for the state duration \cite{rabiner1989}.

Given a set of training sequences, a key inference problem in HSMMs is to compute the probability of a given test sequence. Traditionally, such inference problems have been solved using a two-stage approach: first estimate parameters of the HSMM from the training set, and then use the estimated parameters to compute the probability of the test set. Since the introduction of HSMMs in the 1980s, several methods for estimating the HSMM parameters have been proposed, including the initial idea due to Rabiner \cite{rabiner1989} based on modifications of the Baum-Welch algorithm \cite{baum1970}, the forward-backward algorithm of Yu et al. \cite{yu2002}, etc. However, these algorithms are variants of the expectation maximization (EM) framework \cite{dempster1977} which, in general, has no guarantees in estimating the parameters correctly, and can suffer from slow convergence.

Approaches based on hierarchical Dirichlet processes have also been proposed for HMM \cite{beal2003} and HSMM \cite{mandt2015}, which are the nonparametric Bayesian models avoiding the need to specify the size of the latent space and learn it from data. However, the accuracy of such algorithms is usually sensitive to initialization and may suffer from slow convergence.

In recent years, there has been an increased interest in the spectral algorithms, which provide computationally efficient, local-minimum-free, provably consistent inference and parameter estimation algorithms for latent variable models. For example, Anandkumar et al. \cite{anandkumar2012} 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. This method is based on a tensor decomposition of certain low order moments, computable directly from data, in order to extract the model parameters.
In many problems, however, the end goal is not the recovery of model parameters but the problem of statistical inference. Therefore, we would want to avoid the parameter estimation step and directly compute the quantity of interest. In this regard, Hsu et al. \cite{10} have proposed an efficient spectral algorithm for inference in HMM. It is based on the idea of expressing the model in an observable representation, which does not depend on the model parameters and uses easily computable second and third order sample moments to perform inference. However, their approach was specific to HMM topology and it is not straightforward to generalize it to other latent variable graphical models. Parikh et al. \cite{18} then introduced a spectral algorithm to perform inference in latent tree graphical models with arbitrary topology, and later in \cite{17} a general spectral inference framework for latent junction trees.

In this paper, we utilize the framework of \cite{17} and introduce a novel spectral algorithm for inference in HSMM. Since we address a more specific problem than \cite{17}, the results of our work shed significantly more light into the details of the spectral framework. Specifically, we show that the probability of any test sequence can be written as a logarithmic dependence of the tensor order on maximum length of latent state persistence. Further, the proposed algorithm only relies on a few spectral decompositions and some sparse tensor multiplications, and is quite efficient. We present promising experimental results on synthetic and real data, illustrating the accuracy and efficiency of the proposed algorithm.

The rest of the paper is organized as follows: We briefly introduce notation in Section 2. In Section 3, we present HSMM inference from a tensor product perspective and in Section 4 we introduce the spectral algorithm for inference. In Section 5, we present a careful technical analysis to establish logarithmic dependence of the tensor order on maximum latent state persistence. We present experimental results in Section 6 and conclude in Section 7.

## 2 Notation and Preliminaries

In this section, we cover the basic facts about the tensor algebra, a detailed tutorial on tensors can be found in \cite{12} or \cite{13}. A tensor is defined as a multidimensional array of data, which will be denoted by boldface Euler script letters, e.g., $\mathbf{X} \in \mathbb{R}^{I_{m_1} \times \cdots \times I_{m_N}}$, 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 a corresponding capital boldface letter, e.g., $X^{sp_{1}} \in \mathbb{R}^{I_{m_1} \times \cdots \times I_{m_N}}$.

Any tensor can be matrisized (or flattened) into a matrix. This mapping can be done in multiple ways, the only requirement is that the number of elements is preserved and the mapping is one-to-one. If we split the modes into two disjoint sets, one corresponding to rows and the other to columns, e.g., \{1, \ldots, m_N\} = \{p_1, \ldots, p_K\} \cup \{q_1, \ldots, q_L\}, then a matrisization of $\mathbf{X}$ is denoted by a corresponding capital boldface letter, e.g., $X^{p_1 \cdots p_K q_1 \cdots q_L} \in \mathbb{R}^{I_{p_1} \cdots I_{p_K} \times I_{q_1} \cdots I_{q_L}}$.

Multiplication of two tensors is performed along specific modes. For this, we flatten each tensor to a matrix, perform the usual matrix multiplication and transform the result back to a tensor. The multiplication is denoted by a symbol $\times$ with an optional subscript representing the modes along which the operation is performed, e.g.,

$$
\mathbf{Z}^{p_1 \cdots p_K q_1 \cdots q_L} = \mathbf{X}^{p_1 \cdots p_K q_1 \cdots q_L} \times_{q_1 \cdots q_L} \mathbf{Y}^{p_1 \cdots p_K q_1 \cdots q_L},
$$

where $\mathbf{Z}^{p_1 \cdots p_K q_1 \cdots q_L} \in \mathbb{R}^{I_{q_1} \cdots I_{q_L} \times I_{p_1} \cdots I_{p_K} \times I_{q_1} \cdots I_{q_L}}$ and the resulting tensor on the left hand side is of the form $\mathbf{Z}^{p_1 \cdots p_K q_1 \cdots q_L} \in \mathbb{R}^{I_{q_1} \cdots I_{q_L} \times I_{p_1} \cdots I_{p_K} \times I_{q_1} \cdots I_{q_L}}$. Observe that in the above we can flatten the tensors $\mathbf{X}$ and $\mathbf{Y}$ in multiple different ways as long as the matrix multiplication remains valid. For example, we could assign the multiplication modes in both tensors to columns, in this case the matrix product becomes $\mathbf{Z} = \mathbf{X} \mathbf{Y}^T$. Alternatively, the tensor $\mathbf{Y}$ could be matrisized with the multiplication modes corresponding to rows, resulting in the product $\mathbf{Z} = \mathbf{X} \mathbf{Y}$.

An important fact about tensor multiplication is that in a series of tensor multiplications the order is irrelevant as long as the multiplication is performed along the matching modes, e.g.,

$$
\mathbf{X}^{sp_{1}} \times_{s} (\mathbf{Y}^{tr_{1}} \times_{r} \mathbf{Z}^{tr_{2}}) = (\mathbf{X}^{sp_{1}} \times_{s} \mathbf{Z}^{tr_{2}}) \times_{r} \mathbf{Y}^{tr_{1}}.
$$

If we let the matrisized tensors to be $\mathbf{X} \in \mathbb{R}^{I_{p} \times I_{s}}$, $\mathbf{Y} \in \mathbb{R}^{I_{r} \times I_{p}}$ and $\mathbf{Z} \in \mathbb{R}^{I_{r} \times I_{s}}$, then the above can be verified to be true since

$$
\mathbf{X} (\mathbf{Y} \mathbf{Z}) = (\mathbf{X} \mathbf{Z}^{T}) \mathbf{Y}^{T}.
$$
Note that to reduce clutter, in many places we will drop the multiplication subscripts. The implied modes of multiplication can then be inferred from the subscripts of the tensors. Specifically, when two tensors are multiplied, we first check their modes and then multiply along the modes which are common to both of them. For example, in the product $\mathbf{X} \times \mathbf{Y}$ the implied multiplication is performed along the common modes, i.e., $q$ and $r$.

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

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

where the inversion is performed along the modes $q_1, \ldots, q_L$. To perform tensor inversion, we first matrise it. The modes to be inverted along, are associated with columns of the matrix in which case we would compute the right matrix inverse, so that these modes get eliminated after the product. Alternatively, if the inversion modes are associated with rows of the matrix, then we would compute left matrix inverse. Obviously, for the full rank square matrices both choices would produce the same result. For example, in the above equation the matrisized tensor might be of the form $\mathbf{X}_{p_1, \ldots, p_K, q_1, \ldots, q_L} \in \mathbb{R}^{l_1 \cdots l_K \times l_1 \cdots l_L}$, therefore, we would compute the right matrix inverse so that the modes $q_1, \ldots, q_L$ are eliminated. If the matrisized $\mathbf{X}$ has full row rank, then the inverse can be computed, otherwise we could only compute its pseudo-inverse. Tensorizing the matrix $\mathbf{X}^{-1}$ gives us the desired tensor inverse.

Observe that in the above, $\mathbf{J}_{p_1, \ldots, p_K, q_1, \ldots, p_K}$ has duplicate modes and denotes an identity tensor, whose elements are everywhere zero, except $\mathbf{J}(i_1, \ldots, i_K, i_1, \ldots, i_K) = 1$. In general, if a tensor has duplicate modes, the corresponding sub-tensor can be interpreted as a hyper-diagonal. For example, if for a tensor $\mathbf{X}$ we construct a tensor $\mathbf{X}^{p_{pq}}$, which has its mode $p$ duplicated three times, then for a fixed index $i$, the sub-tensor $\mathbf{X}(i, \ldots, i, i)$ is a hypercube with elements $\mathbf{X}(i, \ldots, i)$ on the diagonal.

Mode duplication enables us to multiply several tensors along the same mode. For example, if we need to multiply tensors $\mathbf{X}_{sp}, \mathbf{Y}_{pr}$ and $\mathbf{Z}_{tp}$ along the mode $p$, then a simple product of the form $\mathbf{X}_{sp} \times_p \mathbf{Y}_{pr} \times_p \mathbf{Z}_{tp}$ cannot be done since any product of two tensors along the mode $p$ would eliminate it, preventing any further multiplications. In general, if there are $N$ multiplications along the specific mode, then there are must be cumulatively $2N$ modes in the participating tensors. In our example, we might duplicate the mode $p$ in, say, tensor $\mathbf{Z}$ to have $\mathbf{X}_{sp} \times_p \left( \mathbf{Y}_{pr} \times_p \mathbf{Z}_{tp} \right)$.

To reduce clutter, in this paper we do not explicitly show the duplicated variables in the subscripts; the implied mode repetition will be evident from the context or explicitly stated in cases when there is a confusion. For example, the notation for the identity tensor becomes $\mathbf{J}_{p_1, \ldots, p_K}$.

### 3 Problem Formulation

In this paper, we consider the problem of inference in HSMM[1] (see Figure 1). Unlike the popular HMM, which has a geometric probability for state persistenced, i.e., the probability of persisting in the same state over $t$ time steps decreases as $p^t$, where $p$ is the probability of persistence for one time step, HSMM explicitly models state persistence. 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

---

1Note: to reduce clutter, in the main part of the paper we ignore the initial and final steps of the model. These details for these parts are presented in Appendix A.
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 \(p(a, 0)\) 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 \(d_t|x_t\). 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 HSMM model, where each sequence is \(S^i = \{o_i^1, \ldots, o_i^{T_i}\}, i = 1, \ldots, N\), our goal is to compute the CPTs using the EM algorithm, and use the estimates to compute \(p(S^\text{test})\). A traditional approach would be to estimate the CPTs using the EM algorithm, and use the estimates to compute \(p(S^\text{test})\). However, the EM algorithm is not guaranteed to estimate the parameters optimally, and hence the computation of \(p(S^\text{test})\) may be incorrect. The focus of our work is to develop a provably correct spectral algorithm for computing \(p(S^\text{test})\).

### 3.1 HSMM in Tensor Notations

We start by considering the matrix forms of the HSMM 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 \(D \in \mathbb{R}^{n_d \times n_x}\), \(p(x_t|x_{t-1}, d_{t-1} = 1)\) is denoted as \(X \in \mathbb{R}^{n_x \times n_x}\), and \(p(o_t|x_t)\) as \(O \in \mathbb{R}^{n_x \times n_o}\). We make the following assumptions on the HSMM parameters:

**Assumptions 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, as a consequence, \(n_x \leq n_o\).

We provide some comments on the above assumptions. We note that the statement 1 in Assumptions 1 can be relaxed to allow zero entries (while still ensuring full rank structure) and thus prevent certain states to be reached from other states, however, this would require more analysis involving Markov chain mixing times of hidden states and is not pursued in this work. Also, observe that the assumption of \(n_x \leq n_o\) is needed in order to ensure that hidden states are identifiable. Intuitively, it means that each hidden state would emit enough observations, so that it can be differentiated from other hidden states.

To express the joint probability \(p(o_1, \ldots, o_T)\) for any possible observation sequence in tensor form, we utilize the junction tree [5] (see Figure 2) corresponding to the graphical model in Figure 1. Recall, that the junction tree is a tree-structured representation of an arbitrary graph enabling efficient computation of the inference problems. It can be constructed by forming a maximal spanning tree from the cliques of the graph. The cliques then represent vertices in

Figure 1: Hidden Semi-Markov Model (HSMM). Here \(o_t\) denotes an observation at time step \(t\), \(x_t\) is a latent state and \(d_t\) is the length of state persistence at time step \(t\). See text for more details.
In this section we present the details of the derivation for the spectral inference approach and provide a practical algorithm implementing these ideas.

### 3.2 Summary of Results

In this work, we represent expression (3) in the observable form so that all the factors can be estimated directly from data using certain sample moments without knowledge of model parameters. The first representation which we derived in Section 4.3 would require estimating $X$, $D$ and $O$ for all time stamps $t$. However, exploiting the homogeneity property of the HSMM, i.e., the fact that the above tensors are time invariant, we derived computationally more efficient and accurate algorithm in Section 4.4 requiring estimation of only three tensors. In Section 5 we study the conditions under which such representation exists. In particular, our analysis shows that the order of the required sample moments has logarithmic dependence on the longest state persistence $n_d$. Finally, we evaluated the proposed algorithm using synthetic and real datasets and compared its performance with the traditional EM approach.

### 4 Spectral Algorithm for Inference in HSMM

In this section we present the details of the derivation for the spectral inference approach and provide a practical algorithm implementing these ideas.

---

Figure 2: Junction Tree for Hidden Semi-Markov Model. The ovals represent cliques, which are denoted by capital blackboard bold variables; the rectangles denote separators. Symbols within the shapes represent the variables on which the corresponding potentials depend.
4.1 Observable Tensor Representation

Observe that the computation of the joint probability in (3) requires knowledge of 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 [17] and between every two factors in (3) introduce an identity tensor with the modes corresponding to the modes along which the multiplication is performed. For example, consider a part of (3) after introducing identity tensors:

\[
\times \mathcal{D}_{x_{t-1}d_{t-2} \omega_{x_{t-1}d_{t-2}}} \times x_{t-1}d_{t-2} \omega_{x_{t-1}d_{t-2}} \times x_{t-1}d_{t-2} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1},
\]

where all the identity tensors have duplicated modes but are not shown.

Now rewrite each of the identity tensors in (4) as a multiplication of some factor times its inverse. For example, \( \mathcal{J} = \mathcal{F} \times \omega_{x_{t-1}d_{t-1}} \mathcal{F}^{-1} \), for some invertible factor \( \mathcal{F} \), whose modes are \( x_t \) and \( \omega_{x_{t-1}d_{t-1}} \). Note that the choice of mode \( x_t \) is fixed and is determined by the modes of the identity tensor \( \mathcal{J} \), while the mode \( \omega_{x_{t-1}d_{t-1}} \) is not fixed and we have a freedom in selecting it. Moreover, observe that the tensor inversion is done along the mode \( \omega_{x_{t-1}d_{t-1}} \), therefore if the matrix \( \mathcal{F} \) has its rows associated with mode \( \omega_{x_{t-1}d_{t-1}} \), we need to ensure such a matrix has full column rank for the inverse to exist and for the product \( \mathcal{F}^{-1} \mathcal{F} \) to equal identity matrix (see Section 2 for more details on tensor inversion). Based on the above discussion, we choose tensor \( \mathcal{F} \) such that (i) the set of variables \( \omega_{x_1} \) are the observed variables, (ii) the resulting factor \( \mathcal{F} \) is invertible and (iii) we interpret this factor as corresponding to a conditional probability distribution, i.e., \( p(\omega_{x_1} | x_1) \) and therefore write \( \mathcal{F} \).

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 three modified tensors:

\[
\tilde{\mathcal{D}}_{\omega_{x_{t-1}d_{t-2}} \omega_{x_{t-1}d_{t-1}}} = \mathcal{F}^{-1} \times x_{t-1}d_{t-2} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1},
\]

\[
\tilde{\mathcal{X}}_{\omega_{x_{t-1}d_{t-1}} \omega_{x_{t-1}d_{t-1}}} = \mathcal{F}^{-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1},
\]

\[
\tilde{\mathcal{O}}_{\omega_{x_{t-1}d_{t-2}} \omega_{x_{t-1}d_{t-1}}} = \mathcal{F}^{-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1} \times x_{t-1}d_{t-1}.
\]

Note that although the above tensors depend 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 \( \mathcal{F}^{-1} \)). For example, \( \mathcal{D} \) is not a tensor form of \( p(\omega_{x_{t-1}d_{t-2}}, \omega_{x_{t-1}d_{t-1}}) \).

Next, we discuss the choice of the observable set \( \omega \) in the factors \( \mathcal{F} \). From Figure 2 we can see that there are three types of separators which depend on \( x_{t-1}d_{t-1} \), \( x_{t-1}d_{t-1} \) and \( x_t \), consequently, there are three types of identity tensors which we introduced in (4), i.e., \( \mathcal{J} \), \( \mathcal{F} \) and \( \mathcal{I} \). Therefore, we need to define three types of observable sets \( \omega_{x_{t-1}d_{t-1}} \), \( \omega_{x_{t-1}d_{t-1}} \) and \( \omega_{x_t} \). There could be multiple choices for these sets, one of them is \( \omega_{x_{t-1}d_{t-1}} = \omega_{x_{t-1}d_{t-1}} = \{o_{t+1}, o_{t+2}, \ldots \} \) for all \( t \) (see Figure 3 for an illustration). The detailed description of how and what number of these observations to select is deferred until Section 5 where we also show that we can set \( \omega_{x_t} = o_t \).

In what follows, we define \( O_{R_t} := \{o_{t+1}, o_{t+2}, \ldots \} \), to emphasize that this is a set of observations starting at time stamp \( t \) and going to the right (or forward in time) in graphical model in Figure 1. With these definitions, we can now rewrite (4) in the form:

\[
P = \prod_{t} O_{R_t} \times \tilde{\mathcal{D}}_{\omega_{x_{t-1}d_{t-2}}} \times \tilde{\mathcal{X}}_{\omega_{x_{t-1}d_{t-1}}} \times \tilde{\mathcal{O}}_{\omega_{x_{t-1}d_{t-1}}}.
\]

Comparing (3) and (5) we see that the above equation expresses the joint probability distribution in the observable form. In what follows, we show how to estimate such tensors directly from data, without the need of the model parameters.
where

\begin{align}
D_{\mathbf{O}_{L_t-1}, \mathbf{O}_{R_t}} = \mathbb{F}^{-1}\mathbf{O}_{L_t-1|x_t-1, d_{t-2}} \times x_{t-1} d_{t-2} \times \mathbb{D}_{x_{t-1} d_{t-2}} \times \mathbf{O}_{R_t|x_{t-1}, d_{t-2}} \mathbb{F},
\end{align}

whose modes are the observable variables \(\mathbf{O}_{R_t-1}\) and \(\mathbf{O}_{R_t}\). To estimate this tensor from data, consider \(\mathbf{O}_{L_t-1}\), a set of the observed variables such that \(\mathbf{O}_{L_t-1}\) and \(\mathbf{O}_{R_t-1}\) are independent, conditioned on \(x_{t-1} d_{t-2}\) (see Figure 3):

\[ p(\mathbf{O}_{L_t-1}, \mathbf{O}_{R_t-1}) = \sum_{x_{t-1} d_{t-2}} p(\mathbf{O}_{L_t-1} | x_{t-1} d_{t-2})p(\mathbf{O}_{R_t-1} | x_{t-1} d_{t-2})p(x_{t-1} d_{t-2}). \]

Write this conditional independence relationship in tensor form:

\[ \mathcal{M}_{\mathbf{O}_{L_t-1}, \mathbf{O}_{R_t-1}} = \mathbf{O}_{L_t-1|x_t-1, d_{t-2}} \times x_{t-1} d_{t-2} \mathbf{O}_{R_t-1|x_{t-1}, d_{t-2}} \times x_{t-1} d_{t-2} \mathcal{K}, \]

where tensor \(\mathcal{K}\) 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 \(\mathcal{K}\), since it interacts with both other terms. The set \(\mathbf{O}_{L_t-1}\) is defined in a way similar to \(\mathbf{O}_{R_t}\), but with the set of observations starting at time stamp \(t - 2\) and going to the left (or backward in time), i.e., \(\mathbf{O}_{L_t-1} := \{\ldots, o_{t-3}, o_{t-2}\}\).

In what follows, we express the inverse of the tensor \(\mathbb{F}_{\mathbf{O}_{R_t-1}|x_{t-1}, d_{t-2}}\) from the above equation and substitute back to (6).

For this, we observe that in (6) the tensor \(\mathbb{F}^{-1}\) is inverted with respect to mode \(\mathbf{O}_{R_t-1}\), therefore, we do the following:

\[ \mathcal{M}_{\mathbf{O}_{L_t-1}, \mathbf{O}_{R_t-1}}^{-1} = \mathbf{O}_{L_t-1|x_{t-1}, d_{t-2}} \times x_{t-1} d_{t-2} \mathbf{O}_{R_t-1|x_{t-1}, d_{t-2}} \times x_{t-1} d_{t-2} \mathcal{K}, \]

where \(\mathcal{M}_{\mathbf{O}_{L_t-1}, \mathbf{O}_{R_t-1}}^{-1}\) is inverted with respect to mode \(\mathbf{O}_{L_t-1}\). Next, substitute (8) back to (6) to get

\begin{align}
\mathbb{D}_{\mathbf{O}_{L_t-1}, \mathbf{O}_{R_t}} &= \mathcal{M}_{\mathbf{O}_{L_t-1}, \mathbf{O}_{R_t-1}}^{-1} \times \mathbf{O}_{L_t-1|x_{t-1}, d_{t-2}} \times x_{t-1} d_{t-2} \mathcal{K}\times x_{t-1} d_{t-2} \mathbb{D}_{x_{t-1} d_{t-2}} \times \mathbf{O}_{R_t|x_{t-1}, d_{t-2}} \mathbb{F},
\end{align}

where

\begin{align}
\mathcal{M}_{\mathbf{O}_{L_t-1}, \mathbf{O}_{R_t-1}}^{-1} = \mathcal{M}_{\mathbf{O}_{L_t-1}, \mathbf{O}_{R_t-1}}^{-1} \times \mathbf{O}_{L_t-1|x_{t-1}, d_{t-2}} \times x_{t-1} d_{t-2} \mathcal{K}\times x_{t-1} d_{t-2} \mathbb{D}_{x_{t-1} d_{t-2}} \times \mathbf{O}_{R_t|x_{t-1}, d_{t-2}} \mathbb{F}.
\end{align}
where we eliminated all the latent variables by multiplying the last four terms on the first line.

Observe that the tensors $\mathbf{O}_{t+1} \mathbf{M}_t$ and $\mathbf{M}_t$ represent valid probability distributions and though they are defined with respect to unknown model parameters, we can readily estimate them from data. For example, $\mathbf{M}_t$ 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 this symbols will be determined in Section 5.

### 4.2.2 Computation of Tensor $\mathbf{O}_{R_t} \mathbf{O}_{R_t}$

The form of this tensor was established to be:

$$
\mathbf{X} = \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right) \times \left( \mathbf{M}_t \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right) \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t} \right) \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right) \times \left( \mathbf{M}_t \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right) \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t} \right).
$$

Consider the following conditional independence relationship (see Figure 3):

$$
\mathbf{K} = \mathbf{K}, \quad \text{and we omitted the duplicated modes.}
$$

Express the inverse of tensor $\mathbf{F}$ from the above equation

$$
\mathbf{F}^{-1} = \mathbf{M}_t \times \mathbf{O}_{R_t} \times \mathbf{F} \times \mathbf{O}_{R_t}^{-1} \times \mathbf{K},
$$

where tensor $\mathbf{O}_{R_t}^{-1}$ is inverted with respect to mode $\mathbf{O}_{R_t}$, while $\mathbf{M}_t$ is inverted with respect to mode $\mathbf{O}_{L_t}$.

Substituting back to (10), we get

$$
\mathbf{X} = \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right) \times \left( \mathbf{M}_t \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right) \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t} \right) \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right) \times \left( \mathbf{M}_t \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right) \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t} \right).
$$

Multiplying together the last five factors, we obtain

$$
\mathbf{X} = \mathbf{M}_t \times \mathbf{O}_{R_t} \times \mathbf{F} \times \mathbf{O}_{R_t} \times \mathbf{K} \times \mathbf{X}_d \times \mathbf{O}_{R_t}^{-1} \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t} \right) \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right) \times \left( \mathbf{M}_t \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right) \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t} \right).
$$

Finally, (10) can now be written as

$$
\mathbf{X} = \mathbf{M}_t \times \mathbf{O}_{R_t} \times \mathbf{M}_t \times \mathbf{O}_{R_t}^{-1} \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t} \right) \times \left( \mathbf{O}_{R_t} \mathbf{O}_{R_t}^{-1} \right),
$$

where the right hand side can now be estimated directly from data, without the need of the model parameters.
4.2.3 Computation of Tensor $\tilde{O}_{t+1}$

Finally, we consider the tensor

$$\tilde{O}_{t+1} = F^{-1}_{t+1} \times_{t+1} O_{t+1} \times_{t+1} O_{t+1}$$  \hspace{1cm} (13)

The conditional independence relationship can take the form

$$M = F_{t+1} \times_{t+1} F_{t+1} \times_{t+1} K$$  \hspace{1cm} (14)

Expressing the inverse of $\mathcal{F}_{t+1}$

$$F^{-1}_{t+1} = M_{t+1}^{-1} \times_{t+1} F_{t+1} \times_{t+1} K \times_{t+1} O_{t+1}$$

and substituting in (13), we get

$$\hat{O}_{t+1} = M_{t+1}^{-1} \times_{t+1} \mathcal{F}_{t+1} \times_{t+1} \mathcal{K} \times_{t+1} \mathcal{O}_{t+1}$$  \hspace{1cm} (15)

4.3 Basic Version of Spectral Algorithm

In the previous section, we expressed the tensors $\mathcal{D}$, $\mathcal{X}$ and $\mathcal{O}$ in terms of the moments directly computable from data. For example, in (15) to estimate $M_{t+1}$, we use a given dataset to compute a matrix $M$ where each entry is $M_{t+1} = \hat{p}(o_t = i, o_{t+1} = j)$ for all $t$ and $\hat{p}(\cdot)$ is the sample estimate of the marginal probability of the observed subsequence.

Using (9), (12) and (15) in (5) we can obtain the spectral algorithm to compute $F_{t+1}$ entirely using the observed variables and Algorithm 1 shows its basic version. Figure 4 shows the graphical representation of this algorithm in terms of the transformed junction tree of Figure 2.

The basic version of the spectral HSMM algorithm can be described as a two step process: in the training step, compute $\mathcal{O}_{R_t} \cdot \mathcal{X}$ and $\mathcal{O}$ for all $t$ using (9), (12) and (15) from the training data. In the testing step, use (5) to compute $p(S_{test})$.

The notation $\tilde{O}_{t+1} | o_t = o_{test}$ means that based on the value of the $t$th symbol in testing sequence, we slice the tensor $\tilde{O}_{t+1}$ along the element $o_{test}^t$ in the dimension $o_t$. For example, if $\tilde{O}_{t+1} \in \mathbb{R}^{10 \times 10}$ and $o_{test}^t = 3$ then $\tilde{O}_{t+1} | o_t = o_{test}^t \in \mathbb{R}^{10 \times 1}$, a third column in the original matrix.

Analyzing (9), (12) and (15), we see that the computational complexity of the training phase of the algorithm is determined by the tensor inverses and multiplications. For example, if in (9) we denote $|O_R| = |O_L| = \ell$, then $M_{R_t} \in \mathbb{R}^{n_t \times n_t^r}$ and $M_{L_t} \in \mathbb{R}^{n_t \times n_t^l}$. The computational complexity of the multiplications and inversions would then be $O(n_t^{2\ell})$. Performing this computations for all $t$ and assuming that the length of training and testing sequences is $T$, would result in $O(n_{test}^{2\ell}T)$. Additionally, there will be a cost of $O(\ell N T)$ to estimate the sample moments $\mathcal{M}$, which is based on counting the co-occurrences of certain observable symbols. Here $N$ is the number of training sequences. In the testing phase of the algorithm, we perform a series of tensor multiplications with the cost of $O(n_{test}^{2\ell}T)$. Thus, the overall cost of Algorithm 1 is then $O((n_{test}^{2\ell} + \ell N)T)$.

In the following section we show how to improve the accuracy and efficiency of the basic spectral Algorithm 1. The idea is to estimate only three tensors $\mathcal{X}$, $\mathcal{D}$ and $\tilde{O}$ in the batch, by averaging across all $t$. 

Algorithm 1 Basic Spectral Algorithm for HSMM inference

**Input:** Training sequences: $S^i = \{o^i_1, \ldots, o^i_T\}$, $i = 1, \ldots, N$.
Testing sequence: $S^{test} = \{o^{test}_1, \ldots, o^{test}_T\}$.

**Output:** $p(S^{test})$

**Training phase:**
for all $t$ do
Estimate $O_{R_{t-1}}$, $O_{R_t}$, $\tilde{X}$ and $\tilde{D}$ from data $\{S^1, \ldots, S^N\}$ using equations (9), (12) and (15).
end for

**Testing phase:**
$p(S^{test}) = 1$
for $t = T$ down to $t = 1$ do
$p(S^{test}) = p(S^{test}) \times \tilde{D} \times O_{R_t} \left( \tilde{X} \times O_{R_{t-1}} \right) \bigg|_{o_t^{test} = o^{test}_t}$
end for

4.4 Efficient Version of Spectral Algorithm

We show the details for computing the tensors $\tilde{D}$ in the batch form. The derivations for other tensors $\tilde{X}$ and $\tilde{D}$ can be computed in a similar manner. Recall from (9) the form of $O_{R_{t-1}}$, and consider the following structure:

\[
\left( \sum_t O_{L_{t-1}} M_{R_{t-1}} \right)^{-1} \times O_L \left( \sum_t O_{L_{t-1}} M_{R_t} \right),
\]

where $O_L$ denotes a generic mode of the averaged tensor $M$, corresponding to $O_{L_{t-1}}$ for all $t$. Note that in practice, instead of summation, we use averaging to avoid numerical overflow problems. It is equivalent to the considered expression, since the term $\frac{1}{T}$ then cancels out.

Since

\[
O_{L_{t-1}} M_{R_{t-1}} = O_{L_{t-1} | x_{t-1} d_{t-2}} \times O_{R_{t-1} | x_{t-1} d_{t-2}} \times O_{R_{t-1} | x_{t-1} d_{t-2}} \times K_{x_{t-1} d_{t-2}},
\]

the first term inside brackets can be rewritten as:

\[
\sum_t O_{L_{t-1} | x_{t-1} d_{t-2}} \times O_{R_{t-1} | x_{t-1} d_{t-2}} \times O_{R_{t-1} | x_{t-1} d_{t-2}} \times K_{x_{t-1} d_{t-2}} = \sum_t O_{R_{t-1} | x_{t-1} d_{t-2}} \times O_{R_{t-1} | x_{t-1} d_{t-2}} \times O_{R_{t-1} | x_{t-1} d_{t-2}} \times K_{x_{t-1} d_{t-2}}
\]

and in the third line we used the homogeneity property of HSMM, i.e., the fact that $O_{R_{t-1} | x_{t-1} d_{t-2}}$ does not depend on time stamp $t$, and extracted one of the common factors. Note that the term $O_{R_{t-1} | x_{t-1} d_{t-2}}$, on the other hand, does depend on $t$ since the factor $K_{x_{t-1} d_{t-2}}$ changes as the time stamp $t$ changes.

Similarly, since

\[
O_{L_{t-1}} M_{R_t} = O_{L_{t-1} | x_{t-1} d_{t-2}} \times O_{R_{t-1} | x_{t-1} d_{t-2}} \times O_{R_{t-1} | x_{t-1} d_{t-2}} \times D_{x_{t-1} x_{t-1} d_{t-2}} \times O_{R_t | x_{t-1} d_{t-1}},
\]

...
Therefore, we can conclude that the batch form of the tensor takes the form:

$$\phi$$

together with the homogeneity property of HSMM.

O terms in parenthesis are invertible. This is due to the fact that the set of observations $O$ rewrite the second term in (16) as

$$\sum_{t} o_{t-1|x_{t-1}d_{t-2}} \times x_{t-1}d_{t-2} \times x_{t-1|x_{t-1}d_{t-2}} \times d_{t-1|x_{t-1}d_{t-2}} \times o_{t|x_{t-1}d_{t-1}} \times F$$

where we used the transformations similar as in (18). Now if we multiply the inverse of (18) with (20), we get

$$\mathcal{F}^{-1} \times \left( \sum_{t} o_{t-1|x_{t-1}d_{t-2}} \times x_{t-1}d_{t-2} \times d_{t-1|x_{t-1}d_{t-2}} \times o_{t|x_{t-1}d_{t-1}} \right) \times \mathcal{D} \times o_{t|x_{t-1}d_{t-1}} \times F \quad (21)$$

where in (21) we used the fact that the order in which tensors are multiplied is irrelevant and also the fact that the terms in parenthesis are invertible. This is due to the fact that the set of observations $O_{t|x_{t-1}}$ for all $t$ is selected so as to make each of the summand invertible (see Section 5 for the details about the choice of $O_{t|x_{t-1}}$). Moreover, in (22) we used the definition of $o_{t-1|x_{t-1}d_{t-2}}$ together with the homogeneity property of HSMM.

Therefore, we can conclude that the batch form of the tensor takes the form:

$$\mathcal{D} = \left( \sum_{t} o_{t|x_{t-1}o_{t-1}} \right)^{-1} \times o_{t|x_{t-1}o_{t}}$$

Similar derivations can be carried out to obtain the rest of the tensors in the batch form:

$$\tilde{X} = \left( \sum_{t} o_{t|x_{t-1}o_{t}} \right)^{-1} \times o_{t|x_{t-1}o_{t}}$$

$$\tilde{O} = \left( \sum_{t} o_{t|x_{t-1}o_{t}} \right)^{-1} \times o_{t|x_{t-1}o_{t}}$$
where in the last expression the mode \( o \) corresponds to the mode \( o_{t+1} \) after averaging of tensor \( \mathcal{M}_{o_{t+1}} \) for all \( t \).

Analyzing (23), (24) and (25), we see that the computational complexity of the training phase of the algorithm is \( \mathcal{O}\left(n_1^{3\ell} + \ell N T\right) \), mainly determined by the tensor additions and the estimation of the sample moments \( \mathcal{M} \). The number of inverses and multiplications is now fixed and independent of sequence length \( T \). The computational complexity of the testing phase is \( \mathcal{O}(n_1^{3\ell} T) \), which is the same as for Algorithm 1. Thus, the overall cost of Algorithm 2 is \( \mathcal{O}\left((n_1^{3\ell} + \ell N) T\right) \).

Note that although not proved, we observed in practice that such a batch tensor computation significantly improves the accuracy of the resulting spectral algorithm. In part, this is due to the fact that we now use more data to estimate the tensors as compared to the original form (5). The estimates obtained in this form have lower variance, which in turn ensures that the inverses we compute in (23), (24) and (25) are more stable and accurate.

The next section addresses the question, which we left unanswered - how and what number of observations to select for sets \( \mathcal{O}_R \) and \( \mathcal{O}_L \).

5   Rank Analysis of Observable Tensors

In Section 4.2.1 when we derived the equations (9), (12) and (15), we glossed over the question of the existence of tensor inverses \( \mathcal{M}_{o_{t+1}}^{-1} \), \( \mathcal{M}_{o_t}^{-1} \) and \( \mathcal{M}^{-1} \). In this section, our task is to analyze the rank structure of these tensors and impose restrictions on the sets \( \mathcal{O}_L \) and \( \mathcal{O}_R \) to ensure that the rank conditions are satisfied. For example, consider equation (9) and expand all its terms using (7) to get

\[
\mathcal{D}_{o_{R_{t-1}} o_{R_t}} = \mathcal{F}_{x_{t+1}d_{t-2}}^{-1} \times \mathcal{K}_{x_{t+1}d_{t-2}}^{-1} \times o_{R_{t-1}} \mathcal{K}_{x_{t+1}d_{t-2}} \times o_{R_{t+1}} \mathcal{F}_{x_{t+1}d_{t-2}} \times \mathcal{D}_{x_{t-1}d_{t-2}} \times \mathcal{F}_{x_{t-1}d_{t-1}},
\]

where we dropped the multiplication subscripts and some of the duplicated modes, which can be inferred from the context. Observe, that in order for the above equation to produce (6), the terms in the middle must multiply out into identity tensor

\[
\mathcal{J}_{x_{t-1}d_{t-2}} = \mathcal{K}_{x_{t+1}d_{t-2}}^{-1} \times o_{R_{t-1}} \mathcal{K}_{x_{t+1}d_{t-2}} \times o_{R_{t+1}} \mathcal{F}_{x_{t+1}d_{t-2}} \times \mathcal{F}_{x_{t-1}d_{t-2}}.
\]

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

\[
\mathcal{J}_{x_{t+1}d_{t-2}} = \mathcal{F}_{x_{t+1}d_{t-2}}^{-1} \times \mathcal{F}_{x_{t-1}d_{t-2}} \times \mathcal{F}_{x_{t-1}d_{t-2}}
\]

therefore, we can conclude that for (9) to exist, the identity statements in (26) and (27) must be satisfied. These statements have implications for the ranks of \( \mathcal{K}_{x_{t-1}d_{t-2}} \) and \( \mathcal{F}_{x_{t-1}d_{t-2}} \) which in turn determine the length of the observation sequences \( \mathcal{O}_{L_{t-1}} \) and \( \mathcal{O}_{R_{t-1}} \).

Since \( \mathcal{K}_{x_{t+1}d_{t-2}} \) represents a distribution \( p(x_{t-1}d_{t-2}) \), its matrisized version is a diagonal matrix with \( p(x_{t-1}d_{t-2}) \) on the diagonal. Using statements 1 and 2 in Assumptions 1 it can be concluded that the diagonal elements in this matrix are non-zero and it has rank \( n_x n_d \), it is thus invertible and so the first equation in (26) is satisfied.

Next, consider the second equation in (26) and recall from Section 2 that if we matrisize the tensor as \( \mathcal{F}_{x_{t+1}d_{t-2}} \in \mathbb{R}^{n_x n_{L_{t-1}} \times n_x n_d} \) then \( \mathcal{F}_{x_{t+1}d_{t-2}} \) must have full column rank \( n_x n_d \) for the proper inverse to exist, implying \( n_x \geq n_x n_d \). Similarly, \( \mathcal{F}_{x_{t-1}d_{t-2}} \) in (27) must have rank \( n_x n_d \). As a consequence of the above, the tensor

\[
\mathcal{M}_{o_{t+1} o_{R_{t+1}}} = \mathcal{F}_{x_{t+1}d_{t-2}} \times \mathcal{F}_{x_{t-1}d_{t-2}} \times \mathcal{K}_{x_{t+1}d_{t-2}}
\]

will have rank \( n_x n_d \) and, in general, is rank-deficient.

The argument above can also be used to show that \( \mathcal{M}_{o_{t+1} o_{R_{t+1}}} \) has rank \( n_x n_d \) since in (12) the tensors \( \mathcal{K}_{x_{t-1}d_{t-2}} \), \( \mathcal{F}_{x_{t-1}d_{t-2}} \) and \( \mathcal{F}_{x_{t+1}d_{t-2}} \) all have rank \( n_x n_d \). Similarly, \( \mathcal{M}_{o_{t+1}} \) will have rank \( n_x \) because in (15) the rank of

12
Then using conditional independence property of graphical model in Figure 1, namely, that the variables $X$ for some tensors $Q$ rank structure of $F$ HSMM transition matrix into tensor where the first equality is due to the homogeneity property of the model and in the second equality we embedded the $X$ Define by $F$ non-sequentially. The rest of Section 5 is devoted to the proof of Theorem 2. We first establish the rank structure of tensor $F$ us to transfer this result for tensors for any $x$ choice was only done to ensure the compactness in our notations, however the HSMM homogeneity property enables the result can be established for all $t$: Theorem 2 Let the number of observations be $|O_{R_{t+1}}| = \ell$ and define the set of indices $S = \{ \max \{ t, t + (n_d - 1) - (n_x^i - 1) \} \mid i = 0, \ldots, \ell - 1 \}$, such that $O_{R_{t-1}} = \{ o_k \mid k \in S \}$ then the rank of tensor $O_{R_{t-1}} = \{ o_k \mid k \in S \}$ is $\min \{ n_x^i, n_x n_d \}$. As a consequence of this result, to achieve the rank $n_x n_d$ we will require $\ell = \left[ 1 + \frac{\log n_d}{\log n_x} \right]$ observations, since we need to ensure $n_x^\ell = n_x n_d$. The span of the selected observations is $n_d$, while their number is only logarithmic in $n_d$. For example, consider the estimation of tensor $M$ for an HSMM with $n_x = 3$ and $n_d = 20$. In this case $\ell = 4$ and $O_{R_t} = \{ o_{t-20}, o_{t-18}, o_{t-12}, o_{t-1} \}$ and $O_{R_t} = \{ o_t, o_{t+1}, o_{t+12}, o_{t+18}, o_{t+20} \}$. Figure 5 illustrates this example. In order to prove the above Theorem, we will focus our analysis on the tensor $O_{R_{t+1}} | x_i d_t$ instead. This specific choice was only done to ensure the compactness in our notations, however the HSMM homogeneity property enables us to transfer this result for tensors for any $t$. Note that $$F_{R_{t+1}} | x_i d_t = \mathcal{X}_{R_{t-1}} | x_i d_t = \mathcal{X}_{R_{t-1}} | x_i d_t \times x_i d_t,$$ where the first equality is due to the homogeneity property of the model and in the second equality we embedded the HSMM transition matrix into tensor $\mathcal{X}$ with mode $d_t$ duplicated. It can be shown that the matricized tensor $x_i d_t \in \mathbb{R}^{n_x n_d \times n_x n_d}$ has rank $n_x n_d$. Therefore, the rank structure of $O_{R_{t+1}} | x_i d_t$ will determine the rank structure of $F_{R_{t+1}} | x_i d_t$, so they are equivalent in this case. The rest of Section 5 is devoted to the proof of Theorem 2. We first establish the rank structure of tensor $O_{R_{t+1}} | x_i d_t$ for sequential set of observations $O_{R_{t+1}}$, and then analyze the rank structure for the observations which were selected non-sequentially.

5.1 Rank Structure of Tensor $O_{R_{t+1}} | x_i d_t$

Define by $X_{R_t} = \{ x_{t+2}, x_{t+3}, \ldots \}$, the sequence of hidden states corresponding to $O_{R_{t+1}} = \{ o_{t+2}, o_{t+3}, \ldots \}$. Then using conditional independence property of graphical model in Figure 1 namely, that the variables $O_{R_{t+1}}$ and $x_i d_t$ are independent given $X_{R_{t+1}}$, we can write:

$$O_{R_{t+1}} | x_i d_t = O_{R_{t+1}} | x_i d_t \times x_i d_t,$$

for some tensors $Q$ and $F$, representing the appropriate probability distributions.
where form can be written as: 

$$V = X R_{t+1}$$

Combining the above conclusion with the fact that the matrized form of the other two tensors in (29) is $F \in \mathbb{R}^{n_x \times n_x n_o}$ and $T \in \mathbb{R}^{n_x \times n_x n_o}$, to ensure invertibility of $F$, we need to select a set of variables $X_{R_{t+1}}$ so that $rank(F) = x_n$. Thus, the problem of the analysis of the rank structure of tensor $O_{R_{t+1}}$ was translated to the problem of rank structure of matrix $X_{R_{t+1}}$. In what follows, we assume that $X_{R_{t+1}} = \{x_t, \ldots, x_{t+\ell}\}$ are sequential and so we would be interested in determining $\ell$ which makes $rank(F) = x_n$. Later, the sequential assumption will be removed and we show how to select such variables in a more efficient way.

5.1.1 Computation of Factor $T$

In order to study the rank structure of $X_{R_{t+1}}$, we will have to understand the mechanism how this matrix is constructed and how the rank changes as the size of $X_{R_{t+1}}$ increases. We start by considering the following conditional independence relationships from the model in Figure 1:

$$p(x_{t+3}, x_{t+2}|x_{t+1}, d_{t+1}) = \sum_{d_{t+2}} p(x_{t+3}|x_{t+2}, d_{t+2}) p(d_{t+2}|x_{t+2}, d_{t+1}) p(x_{t+2}|x_{t+1}, d_{t+1})$$

(30)

$$p(x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t) = \sum_{d_{t+1}} p(x_{t+3}|x_{t+2}, x_{t+1}, d_{t+1}) p(d_{t+1}|x_{t+1}, d_t) p(x_{t+1}|x_t, d_t).$$

(31)

Using the model’s homogeneity property, we see that the quantity underlined in (30) is the same as the one in (31). Moreover, equation (30) can then be thought of as transforming $p(x_{t+1}|x_t, d_t)$ into $p(x_{t+2}, x_{t+1}|x_t, d_t)$, while the expression in (31) is, in effect, transforms $p(x_{t+2}, x_{t+1}|x_t, d_t)$ into $p(x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t)$. Thus (30) and (31) encode the following chain of transformations:

$$p(x_{t+1}|x_t, d_t) \rightarrow p(x_{t+2}, x_{t+1}|x_t, d_t) \rightarrow p(x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t).$$

Based on the above considerations, we can rewrite (30) and (31) in the tensor form as follows:

$$x_{t+3}, x_{t+2}|x_{t+1}, d_{t+1} \quad \Upsilon = \quad x_{t+3}, x_{t+2}|x_{t+2}, d_{t+2} \quad x_{t+2}, d_{t+2}|x_{t+1}, d_{t+1} \quad \Upsilon$$

$$x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t \quad \Upsilon = \quad x_{t+3}, x_{t+2}, x_{t+1}|x_{t+1}, d_{t+1} \quad x_{t+1}, d_{t+1}|x_{t+1}, d_t \quad \Upsilon$$

(32)

(33)

where $x_{t+3}, x_{t+2}, x_{t+1}|x_t, d_t = x_{t+3}, x_{t+2}|x_t, d_t = D x_{t+3}, x_{t+2}|x_t, d_t$. The homogeneity property allows us to rewrite the above as

$$x_{t+3}, x_{t+2}|x_t, d_t = D x_{t+3}, x_{t+2}|x_t, d_t.$$  

(34)

(35)

Our next step is to represent the above tensor equations in the matrix form. First, consider tensor $\Upsilon$, its matricized form can be written as:

$$\Upsilon = \quad \mathbf{D} \quad \mathbf{X}$$

(36)

where $\mathbf{D} \in \mathbb{R}^{n_x n_o \times n_x n_o}$ and $\mathbf{X} \in \mathbb{R}^{n_x n_o \times n_x n_o}$. Next, consider the equations (34) and (35), its matrix version is of the form:

$$\mathbf{T} = \mathbf{T} \quad \mathbf{V}$$

(37)

$$\mathbf{T} = \mathbf{T} \quad \mathbf{V}$$

(38)
In the next section, we will provide analysis of Algorithm 3 and specifically study the rank structure of matrix $T$.

Algorithm 3 summarizes the above constructions for a general case:

**Input:** $p(d_l|x_t, d_t-1)$ and $p(x_t|x_{t-1}, d_{t-1})$ - duration and transition distributions, $\ell$ - number of steps

**Initialization:**

\[
p(x_{t+1}|x_t, d_t) \rightarrow T_{x_{t+1}|x_t, d_t}
\]
\[
p(d_{t+1}|x_t, d_t) \rightarrow D_{x_{t+1}, d_{t+1}|x_t, d_t}
\]
\[
p(x_{t+1}|x_t, d_t) \rightarrow X_{x_{t+1}, d_t|x_t, d_t}
\]
\[
V = D_{x_{t+1}, d_{t+1}|x_t, d_t} X_{x_{t+1}, d_t|x_t, d_t}, \quad E = [I \cdots I]
\]

for $i = 1$ to $\ell - 1$

\[
T_i = T_{x_{i+1}, \ldots, x_{i+1} | x_t, d_t} \ast E\]
\[
T_{i+1} = T_{x_{i+2}, \ldots, x_{i+2}, x_{i+1} | x_t, d_t} \ast V
\]

end for

Here $T_{x_{i+1}, x_t | x_t, d_t} \in \mathbb{R}^{n_x \times n_x n_d}$ and, similarly, $T_{x_{i+2}, x_{i+1} | x_t, d_t} \in \mathbb{R}^{n_x \times n_x n_d}$, and matrix $T_{x_{i+3}, x_{i+2}, x_{i+1} | x_t, d_t} \in \mathbb{R}^{n_x \times n_x n_d}$.

Summarizing the above derivations, we can describe the following algorithmic approach for analyzing $T_{x_{R_t+1} | x_t, d_t}$ as $X_{R_t+1}$ increases. We begin with $T_{x_{R_t+1} | x_t, d_t} = [X \ I \cdots I] \in \mathbb{R}^{n_x \times n_x n_d}$, where the first block $X$ corresponds to $d_t = 1$, and the subsequent $(n_d - 1)$ blocks of $I \in \mathbb{R}^{n_x \times n_x}$ correspond to $d_t > 1$ for which $x_{R_t+1} = x_t$.

We then use (37) to get $T_{x_{R_t+1} | x_t, d_t}$. However, notice that in (37) the matrix $T_{x_{R_t+1} | x_t, d_t}$ has a duplicated mode $x_t$, therefore, we need to restructure $T_{x_{R_t+1} | x_t, d_t}$, which can be accomplished with:

\[
T'_{x_{R_t+1}, x_t | x_t, d_t} = T_{x_{R_t+1}, x_t | x_t, d_t} \ast E,
\]

where $E = [I \cdots I] \in \mathbb{R}^{n_x \times n_x n_d}$ and $\ast$ denotes a Khatri-Rao product (row-wise Kronecker product$^2$).

Then, we use (38) to transform $T_{x_{R_t+1} | x_t, d_t}$ into $T'_{x_{R_t+1} | x_t, d_t}$ where, again a preliminary step is to restructure the matrix as follows:

\[
T'_{x_{R_t+1}, x_t | x_t, d_t} = T_{x_{R_t+1}, x_t | x_t, d_t} \ast E.
\]

Algorithm summarizes the above constructions for a general case.

In the next section, we will provide analysis of Algorithm and specifically study the rank structure of matrix $T$.

To understand the analysis, it is important to know how the structure of matrix $T$ evolves across iterations. For this, we present in Figure a schematic description of a few steps of the algorithm.

---

2 Let $P = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix} \in \mathbb{R}^{m \times n}$ and $Q = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix} \in \mathbb{R}^{k \times n}$ then $P \odot Q = \begin{bmatrix} p_1 \odot q_1 \\ p_2 \odot q_2 \\ \vdots \\ p_n \odot q_n \end{bmatrix} \in \mathbb{R}^{mk \times n}$, where $\odot$ is a Kronecker product.
5.1.2 Analysis of Algorithm 3

In this Section our goal is to analyze the Algorithm 3 and study how the rank of matrix $T^{x}_{Rx+1|xt,dt}$ evolves across iterations. First, we state the main result of this analysis:

**Theorem 3** The rank of the output matrix $T^{x}_{Rx+1|xt,dt}$ in Algorithm 3 is $\min(n_x, n_x n_d)$.

Applying now Theorem 3 to equation (29) in matrix form

$$O_{Rt+1|x_t,dt} = Q_{Rt+1|x_t,dt} \times T^{x}_{Rx+1|xt,dt},$$

where $\text{rank}(Q) = n_x^\ell$, we can now conclude the following result:

**Corollary 4** To achieve the full column rank for $O_{Rt+1|x_t,dt} \in \mathbb{R}^{n_x^\ell \times n_x n_d}$, i.e., to ensure that the rank of tensor $\mathcal{F}$ is $n_x n_d$, the number of observations $\ell$ in $O_{Rt+1}$ = \{o_{t+2}, o_{t+3}, \ldots, o_{t+\ell}\} must be equal to the maximum state persistence i.e., $\ell = n_d$.

Before we prove Theorem 3, we will establish certain auxiliary results.
Lemma 5 Let \( A \in \mathbb{R}^{m \times n} \) be a matrix with no all-zero columns then \( \text{rank} (I \odot A) = \text{rank} (A \odot I) = n \), where \( \odot \) denotes Khatri-Rao product and \( I \in \mathbb{R}^{n \times n} \).

**Proof** Let \( K = (I \odot A) \in \mathbb{R}^{mn \times n} \). By definition of Khatri-Rao product, \( K(:,j) = e_j \odot A(:,j) \), for \( j = 1, \ldots, n \), which consists of zeros, except for rows \( (j - 1)m + 1, \ldots, (j - 1)m + m \), containing the column \( A(:,j) \). Here \( \odot \) denotes Kronecker product and \( e_j \) is everywhere zero except for position \( j \) which is 1. As long as there is no all-zero columns in \( A \), each column of \( K \) is independent of each other and therefore the rank is \( n \). Moreover, since the matrix \( A \odot I \) is a row-permuted version of \( A \odot I \), their ranks are the same.

Lemma 6 Define a block-row matrix \( M = [A_1 \ A_2 \ \cdots \ A_k] \in \mathbb{R}^{n \times kn} \), where each \( A_i \in \mathbb{R}^{m \times n} \). Define by \( r_j, j = 1, \ldots, n \) the rank of matrix \( [A_1(:,j) \ \cdots \ A_k(:,j)] \) composed of \( j \)th columns of \( A \)'s, and let \( E = [I \ I \ \cdots \ I] \in \mathbb{R}^{n \times kn} \), where \( I \in \mathbb{R}^{n \times n} \). Then the rank of matrix \( W = M \odot E \in \mathbb{R}^{mn \times kn} \), obtained using a Khatri-Rao product, is \( \min(mn, \sum_j r_j) \).

**Proof** First note that \( M \odot E \) and \( E \odot M \) are row permuted version of each other, so they have the same rank. Therefore, consider \( W' = E \odot M = [I \odot A_1 \ \cdots \ I \odot A_k] \). Also, note that \( e_j \odot [A_1(:,j) \ \cdots \ A_k(:,j)], j = 1, \ldots, n \) is a matrix which consists of zeros except for rows \( (j - 1)m + 1, \ldots, (j - 1)m + m \) where it contains the columns \( [A_1(:,j) \ \cdots \ A_k(:,j)] \). The rank of these columns is \( r_j \) and all other columns in \( W \) are independent of them due to the structure of the Khatri-Rao product. Therefore, each set of such columns adds \( r_j \) to the total rank. Since the overall rank of \( W \) cannot exceed either the number of rows or columns, we conclude that \( \text{rank}(W) = \min(mn, \sum_j r_j) \).

Lemma 7 Let \( V = \{v_1, \ldots, v_n\} \) be a set of independent vectors. Define \( u = \sum_{i=1}^n c_i v_i \), where coefficients \( c_i \neq 0, i = 1, \ldots, n \). Define \( U \) to be a strict subset of \( V, i.e. U \subset V \), then a set of vectors \( u \cup U \) is independent.

**Proof** Define \( \{1, \ldots, n\} = \alpha \cup \bar{\alpha} \), where \( \alpha \) denotes a subset of indices for vectors corresponding to \( U \). Then we can write \( u = \sum_{i \in \alpha} c_i v_i \) and \( \sum_{j \in \bar{\alpha}} c_j v_j \).

Assuming the opposite, i.e., \( u \cup U \) are dependent, we can write \( k_0 u + \sum_{i \in \alpha} k_i v_i = 0 \) where \( k_0 \neq 0 \) and some of \( k_i, i \in \alpha \) are also must be non-zero. Substituting the definition of \( u \) and rearranging the terms, we get:

\[
k_0 \sum_{i \in \alpha} (c_i + k_i) v_i + k_0 \sum_{j \in \bar{\alpha}} c_j v_j = 0.
\]

Since \( c_j \neq 0, j \in \bar{\alpha} \), the above equation claims the linear dependence of vectors in \( V \), which is a contradiction of our assumption and so \( u \cup U \) are independent.

We now ready to analyze Algorithm [3]. It can be verified that (36) is of the form:

\[
V = \begin{bmatrix}
\Psi & I \\
\vdots & \ddots & I \\
0 & \cdots & 0
\end{bmatrix} \in \mathbb{R}^{n_x n_d \times n_x n_d}, \quad \text{where} \quad \Psi = \begin{bmatrix}
\text{diag} [D(1,:)] \chi' \\
\text{diag} [D(2,:)] \chi' \\
\vdots \\
\text{diag} [D(n_d,:)] \chi'
\end{bmatrix} \in \mathbb{R}^{n_x n_d \times n_x}, \quad (41)
\]

where \( \text{diag} [D(i,:)] \) is the diagonal matrix with \( i \)th row from \( D \) on the diagonal. Note that we can also write \( \Psi = (D \odot I) \chi' \). Observe that the rank of \( V \) is \( n_x n_d \) because the \( n_x (n_d - 1) \times n_x (n_d - 1) \) block diagonal matrix delineated in (41) and the last \( n_x \times n_x \) block matrix \( \text{diag} [D(n_d,:)] \chi' \) in \( \Psi \) together comprising \( n_x n_d \) independent columns of \( V \). Note that \( \text{diag} [D(n_d,:)] \chi' \) has rank \( n_x \) because \( \chi' \) is full rank and \( D(n_d,:) \) is non-zero, which follows from statements 1 and 2 in Assumptions [1]. As a side note observe that the requirement to have \( D(n_d,:) \) non-zero implies that there is a non-zero probability of maximum state persistence.

In analyzing the Algorithm [3] it would be useful to denote the matrices at iteration \( i \) in (39) and (40) as

\[
x_{t+i-1,...,t+i} | x_t, d_t = \begin{bmatrix} A_1^{(i)} & \cdots & A_{n_d}^{(i)} \end{bmatrix} \\
x_{t+i-1,...,t+i} | x_t, d_t' = \begin{bmatrix} B_1^{(i)} & \cdots & B_{n_d}^{(i)} \end{bmatrix} \\
x_{t+i, \ldots, x_{t+i+1}} | x_t, d_t = \begin{bmatrix} C_1^{(i)} & \cdots & C_{n_d}^{(i)} \end{bmatrix}.
\]
Moreover, utilizing the structure of matrix $V$ from (41), the operations involved in step (40) are as follows:

$$\begin{bmatrix} C_1(i) & C_2(i) & C_3(i) & \cdots & C_{n_d}(i) \end{bmatrix} = \begin{bmatrix} B_1(i) & \cdots & B_{n_d}(i) \end{bmatrix} \Psi \begin{bmatrix} B_1(i) & B_2(i) & \cdots & B_{n_d}(i) \end{bmatrix}. $$

(42)

With the above information we can now present the proof of Theorem 3.

**Proof** At the start of the algorithm, we have $T_{x_t+1|x_t\{d\}} = [\mathcal{X} I \cdots I] = [A_1^{(1)} \cdots A_{n_d}^{(1)}]$, which has rank $n_x$. The rank of matrix $[A_1^{(1)}(,l) \cdots A_{n_d}^{(1)}(,l)]$ for $l = 1, \ldots, n_x$ is $r_l = 2$ since among all the columns only two of them are independent. Therefore, according to Lemma 6, the result of operations in (39) has rank $\sum r_l = 2n_x$. Moreover, we note that since $[B_1^{(1)} B_2^{(1)} \cdots B_{n_d}^{(1)}] = [\mathcal{X} \otimes I I \cdots I \otimes I]$, it can be seen that its $2n_x$ independent vectors can be formed by the columns $[B_1^{(1)} B_2^{(1)}]$; so the rank of $[B_1^{(1)}(,l) \cdots B_{n_d}^{(1)}(,l)]$ for $l = 1, \ldots, n_x$ is 2.

Next, since the rank of $V$ is $n_xn_d$, the operations in (40) produce matrix $[C_1^{(1)} C_2^{(1)} \cdots C_{n_d}^{(1)}]$ with the rank still being $2n_x$. Moreover, the columns of $C_1^{(1)}$ are linearly dependent on the rest of the columns, $[C_2^{(1)} \cdots C_{n_d}^{(1)}]$, due to (42). However, the rank of $[C_1^{(1)}(,l) \cdots C_{n_d}^{(1)}(,l)]$ is now $r_l = 3$ for $l = 1, \ldots, n_x$. To understand this, note that

$$\begin{bmatrix} B_1^{(1)} & B_2^{(1)} & \cdots & B_{n_d}^{(1)} \end{bmatrix} = [\mathcal{X} \otimes I I \cdots I \otimes I],$$

and, according to (42), $C_1^{(1)} = [B_1^{(1)} \cdots B_{n_d}^{(1)}] \Psi$. As we established before, the rank of $[C_2^{(1)}(,l) \cdots C_{n_d}^{(1)}(,l)] = [B_1^{(1)}(,l) \cdots B_{n_d}^{(1)}(,l)]$ is $r_l = 2$. Moreover, it can also be checked that $C_1^{(1)}(,l)$ is independent of $[C_2^{(1)}(,l) \cdots C_{n_d}^{(1)}(,l)]$ due to Lemma 7. Clearly, then the cumulative rank of $[C_1^{(1)}(,l) \cdots C_{n_d}^{(1)}(,l)]$ is 3 for $l = 1, \ldots, n_x$.

To generalize, if at the iteration $i$ the rank of $[A_1(i) \cdots A_{n_d}(i)]$ is $in_x$ while the rank of $[A_1(i) \cdots A_{n_d}(i)]$ is $(i + 1)$, then the operations in step (39) produce $[B_1(i) \cdots B_{n_d}(i)]$ having rank $(i + 1)n_x$ due to Lemma 6. The step in (40) keeps the rank of $[C_1(i) \cdots C_{n_d}(i)]$ at $(i + 1)n_x$ due to the full rank structure of $V$. At the same time, this step increases the rank of $[C_1(i) \cdots C_{n_d}(i)]$ to $(i + 2)$ due to Lemma 7, i.e., independence of $C_1(i)$ from $[C_2(i) \cdots C_{n_d}(i)]$ with the latter having the rank $(i + 1)$. Therefore, each iteration increases the rank of matrix $T$ by $n_x$ and so after $2 \leq \ell \leq n_d$ steps the rank of the resulting matrix $x_{n_{t+1}|x_{t}\{d\}}$ is $\ell n_x$.

Note that if $\ell = 1$ then the Algorithm 3 is not executed and returns the trivial $x_{t+1|x_{t}\{d\}}$ with rank $n_x$. On the other hand, if $\ell > n_d$ then the rank of $x_{n_{t+1}|x_{t}\{d\}}$ is $n_xn_d$ since this is the number of columns in that matrix and so is the maximum achievable rank.

**5.2 Efficient Computation of Factor $T$**

In Corollary 4 we established that the required number of observations in $O_{R_{t+1}} = \{o_{t+2}, o_{t+3}, \ldots, o_{t+\ell}\}$ is $\ell = n_d$. Therefore, the sizes of the estimated quantities $\hat{D} \in \mathbb{R}^{n_d \times n_d}$ and $\hat{T} \in \mathbb{R}^{n_d \times n_d \times n_o}$ in the Algorithm 2 will have exponential dependency on $n_d$. When maximum state persistence is large, the estimation of such quantity becomes impractical. Fortunately, we can modify Algorithm 3 to significantly reduce the number of observations. The idea is to apply the step (40) multiple times in-between the applications of step (39). Recall that in the previous construction we established that we needed $\ell = n_d$ consecutive observations, e.g., $O_{R_{t+1}} = \{o_{t+2}, \ldots, o_{t+\ell}\}$. In contrast, in the proposed approach, every time we add an observation, say $o_{t+\gamma}$, we skip certain number $\delta$ of time steps before adding another observation $o_{t+\gamma+\delta}$, so that the observations are non-consecutive. As we illustrate next, the span of these non-consecutive observations is still $n_d$ but the number of them is only logarithmic in $n_d$. The proposed approach still achieves the full rank structure of $O_{R_{t+1}|x_{t}\{d\}}$ but with smaller number of data points. The Algorithm 7 which is a simple modification of the Algorithm 3, summarizes the above procedure. The following result establishes the rank structure of the matrix $x_{n_{t+1}|x_{t}\{d\}}$ in the output of the Algorithm 4.
Algorithm 4 Efficient computation of $T_{Rt+1|x_{t+1}d_t}$

**Input:** $p(d_t|x_t, d_{t-1})$ and $p(x_t|x_{t-1}, d_{t-1})$ - duration and transition distributions, $\ell$ - number of steps

**Initialization:**

\[
p(x_{t+1}|x_t, d_t) \rightarrow T_{x_{t+1}|x_t, d_t}
\]
\[
p(d_{t+1}|x_{t+1}, d_t) \rightarrow D_{x_{t+1}, d_{t+1}|x_{t+1}, d_t}
\]
\[
p(x_{t+1}|x_t, d_t) \rightarrow X_{x_{t+1}, d_{t}|x_t, d_t}
\]
\[
V = D_{x_{t+1}, d_{t+1}|x_{t+1}, d_t} X_{x_{t+1}|x_t, d_t}, \quad E = [I \cdots I]
\]

$c = 1$

for $i = 1$ to $\ell - 1$ do

\[
T = T \cdot V
\]

if $i == (n_x)^c - 1$ or $i == \ell - 1$ do

\[
T = T \odot E
\]

end if

\[
c = c + 1
\]

c = $\text{min}(n_x^n, n_x n_d)$.

Theorem 8 The rank of the output matrix $T_{Rt+1|x_{t+1}d_t}$ in Algorithm 4 is $\text{min}(n_x^n, n_x n_d)$.

Note that based on the above theorem, Algorithm 4 increases the rank at every step exponentially rather than linearly. In order for $T_{Rt+1|x_{t+1}d_t}$ to achieve the rank $n_x n_d$ we will now require $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$ observations, since we need to ensure $n_x^n = n_x n_d$. Observe that the span of the selected observations is still $n_d$, while the number of the observations is only logarithmic in $n_d$. The following Corollary summarizes the above conclusions.

Corollary 9 To achieve the full column rank for $O_{Rt+1|x_{t}d_{t}} \in R^{n_x^n \times n_x n_d}$, i.e. to ensure that the rank of tensor $O_{Rt+1|x_{t}d_{t}}$ is $n_x n_d$, the number of observations $\ell$ in $O_{Rt+1|x_{t}d_{t}}$ must be equal to $\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil$, since we need to ensure $n_x^n = n_x n_d$.

Before we prove Theorem 8, it is instructive to visualize the progress of Algorithm 4. Figure 7 shows a schematic description of a few steps of the algorithm. We are now ready to present the proof of Theorem 8.

**Proof** For the proof, we refer back to Algorithm 3 and the proof of Theorem 3. Recall, that at iteration $i = 1$, the result of step (39) is a matrix $[B_1^{(1)} \cdots B_{n_d}^{(1)}] \in R^{n_x^n \times n_x n_d}$, whose rank is $2n_x$, since $[A_1^{(1)}(:, l) \cdots A_{n_d}^{(1)}(:, l)] = [A^{(1)} \cdots I] \in R^{n_x^n \times n_x n_d}$ for $l = 1, \ldots, n_x$ had two independent columns. Then, the transformations in step (40) produced $[C_1^{(1)}(:, l) \cdots C_{n_d}^{(1)}(:, l)]$ for $l = 1, \ldots, n_x$ with rank $3n_x$.

Note that if $n_x > 2$ then $[A_1^{(1)}(:, l) \cdots A_{n_d}^{(1)}(:, l)]$ potentially can have a rank up to $n_x$, while in Algorithm 3 we only have it equal to 2. It turns out that if we apply step (40) multiple times and use Lemma 7 we can increase the rank of $[C_1^{(1)}(:, l) \cdots C_{n_d}^{(1)}(:, l)]$ for $l = 1, \ldots, n_x$ to $n_x$.
that there is a non-zero probability of maximum state persistence. Note that diag
is the diagonal matrix with
is non-zero, which follows from
block diagonal matrix delineated
This example illustrates the HSMM with $n_e = 5$ and $n_d = 10$. The non-zero matrix elements are displayed as dots.
Specifically, consider the step (43). Then at iteration \( i = 1 \) we have \([A_1^{(1)} \cdots A_{n_d}^{(1)}] = [B_1^{(1)} \cdots B_{n_d}^{(1)}]\) and for \( l = 1, \ldots, n_x \) the two independent columns are \([B_1^{(1)}(:, l) \quad B_2^{(1)}(:, l)] = [X(:, l) \quad I(:, l)]\). The result of step (43) gives us then three independent columns

\[
\begin{bmatrix}
C_1^{(1)}(:, l) & C_2^{(1)}(:, l) & C_3^{(1)}(:, l)
\end{bmatrix} = \begin{bmatrix}
[C_1^{(1)}(:, l) \quad X(:, l) \quad I(:, l)]
\end{bmatrix},
\]

where \(C_1^{(1)} = [X \quad I \cdots \quad I]\Psi\). The independence follows from Lemma 7. The repeated application of step (43) one more time gives four independent columns

\[
\begin{bmatrix}
C_1^{(2)}(:, l) & C_2^{(2)}(:, l) & C_3^{(2)}(:, l) & C_4^{(2)}(:, l)
\end{bmatrix} = \begin{bmatrix}
[C_1^{(1)}(:, l) \quad X(:, l) \quad I(:, l)]
\end{bmatrix},
\]

where \(C_1^{(2)} = [C_1^{(1)} \cdots C_{n_d}^{(1)}]\Psi\). Observe that since the number of rows is \(n_x\), we can increase the rank at most up to \(n_x\). Therefore, if in the beginning we had two independent columns and we want to get \(n_x\) independent columns, we will need to apply the step (43) \(n_x - 2\) times, so we will have \([C_1^{(n_x-2)}(:, l) \cdots C_{n_d}^{(n_x-2)}(:, l)]\) with rank \(n_x\).

If we now apply step (44) it will give us \([A_1^{(1)} \cdots A_{n_d}^{(1)}] \in \mathbb{R}^{n_x \times n_x n_d}\) with rank \(n_x^2\) due to Lemma 3. Continuing in this manner, we can again repeatedly apply the step (43) to create a matrix with a rank at most \(n_x^2\), since there are \(n_x^2\) rows and assuming that \(n_x n_d \geq n_x^2\). The number of times we need to apply (43) is now \(n_x^2 - n_x\) since we need to go from \(n_x^2\) to \(n_x^2\) independent columns.

In general, the step (43) needs to be applied \(n_x^c - n_x^{c-1}\), in order to obtain \(n_x^c\) independent columns. The application of step (44) then creates \(T\) with rank \(n_x^{c+1}\). Note, that since \(T\) has \(n_x n_d\) columns, the maximum achievable rank is \(n_x n_d\).

Observe that the above proof also provided the method for selecting the non-sequential observations \(X_{\mathcal{R}_{t+1}}\). Specifically, since the set of observations \(X_{\mathcal{R}_{t+1}} = \{o_{t+2}, \ldots\}\) must start from observation \(o_{t+2}\) and \(X_{\mathcal{R}_{t+1}} = \ell\), we denote \(s = t + 2\). Then, \(t\)th added observation is \(o_{s+(n_d-1)-(n_x-1)}\) for \(i = 0, \ldots, \ell - 2\) and the \(t\)th observation is \(o_x = o_{t+2}\). For tensor \(X_{\mathcal{R}_{t+1}}\) to achieve rank \(n_x n_d\) we need to add \(\ell = \lceil 1 + \frac{\log n_d}{\log n_x} \rceil\) observations.

Theorem 8 together with Corollary 9 now proves Theorem 2 stated earlier.

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.

![Figure 8: Performance of the spectral algorithm and EM on synthetic data generated from HSMM with \(n_o = 5, n_x = 4, n_d = 6\). Left: Error for EM across different iterations for various training datasets. The straight lines show the performance for spectral method. Center: Average error and one standard deviation over 100 runs for EM after convergence and spectral algorithm across different number of training data. Right: Runtime, in seconds, for both methods.](image)

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 an HSMM with \(n_o = 5, n_x = 4, n_d = 6\). We generated a set of \(N = \{500, 1000, 5000, 10^4, 10^5\}\)
training and \( N = 1000 \) testing sequences, each of length \( T = 100 \). The estimation accuracy for each testing sequence was measured using the relative deviation from the true likelihood, i.e., \( \epsilon = \frac{|p(S^{\text{test}}) - p(S^{\hat{S}^{\text{test}}})|}{p(S^{\text{test}})} \). The final reported error is then the root-mean-square error (RMSE) across all the 1000 testing sequences. Figure 8 shows the evaluation results.

It can be observed from the middle panel of Figure 8 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 the EM. We also observed 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, EM achieves lower errors with smaller number of training sets; since it has no global optimum guarantees, its accuracy does not improve much with more data as it gets trapped in the local minima. The right panel in Figure 8 shows that spectral method is several orders of magnitude faster than EM. Therefore, 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 demands smaller computational resources.

6.2 Real Data

We also evaluated the performance of the spectral algorithm on real dataset. For this purpose, we considered a problem of anomaly detection in the civilian aviation systems \([7, 14]\) and used HSMM to detect abnormal flights based on pilot actions. Consider, for example, a part of the flight related to approach, where a pilot goes through a set of phases and uses various controls to decrease the altitude. Using HSMM, we model the phases of the approach part as hidden states and the pilot actions are the observations from these phases. A detailed description of how HSMM can be used to detect aviation safety incidents can be found in \([15]\).

For model evaluation we used NASA flight dataset \([1]\), containing flight information of 35 aircrafts from a defunct Midwest airline company. For each flight, the data has a record of 186 parameters, including sensor readings and pilot actions. We considered a part of flight related to approach for a subset of flights landing at the same airport (see right panel in Figure 9 for an example). 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 10,020 normal flights for training. A testing 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, i.e., from tails of the constructed histogram. Most of the 100 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 (see left panel of Figure 9 for an example).

We applied the spectral algorithm to compute the normalized joint log-likelihood of the observed pilot actions. The high-likelihood sequences were considered normal and low-likelihood sequences were classified as anomalous (see center panel in Figure 9). The algorithm learnt the observable representation model in 189 seconds and achieved a high classification accuracy, which can be observed on the right panel, showing the ROC curve with Area Under Curve (AUC) score of 0.96.
7 Conclusion

In this paper, we present a novel spectral algorithm to perform inference in HSMM. We derive an observable representation of the model which can be computed from the data sample moments of size logarithmic in the maximum latent state persistence. Based on the representation, we present an efficient approach to inference, which relies on tensor operations and a few SVD decompositions. Moreover, the 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.

Acknowledgments

This work was supported by NASA grant NNX12AQ39A, and NSF Grants IIS-0953274, IIS-1029711, IIS-0916750. We thank Nikunj Oza and Bryan Matthews at NASA for their helpful comments and suggestions, and computing support from the Minnesota Supercomputing Institute (MSI). A. B. acknowledges support from IBM and Yahoo.

References

[1] NASA Flight Dataset. Available at https://c3.nasa.gov/dashlink/projects/85/.
[2] A. Anandkumar, R. Ge, D. Hsu, and S. M. Kakade. A tensor spectral approach to learning mixed membership community models. In Conference on Learning Theory, 2013.
[3] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky. Tensor decompositions for learning latent variable models. arXiv preprint arXiv:1210.7559, 2012.
[4] A. Anandkumar, A. Javanmard, D. Hsu, and S. M. Kakade. Learning linear bayesian networks with latent variables. In Proceedings of the International Conference on Machine Learning, volume 28, pages 249–257, 2013.
[5] D. Barber. Bayesian Reasoning and Machine Learning. Cambridge University Press, 2012.
[6] L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. The Annals of Mathematical Statistics, 37(6):1554–1563, 1966.
[7] S. Das, B. Matthews, A. Srivastava, and N. Oza. Multiple kernel learning for heterogeneous anomaly detection: Algorithm and aviation safety case study. In Proceedings of the International Conference on Knowledge Discovery and Data Mining, pages 47–56, 2010.
[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society, pages 1–38, 1977.
[9] E. B. Fox., E. B. Sudderth, M. I. Jordan, and A. S. Willsky. An HDP-HMM for systems with state persistence. In Proceedings of the 25th International Conference on Machine Learning, pages 312–319, 2008.
[10] D. Hsu, S. M. Kakade, and T. Zhang. A spectral algorithm for learning hidden Markov models. Journal of Computer and System Sciences, 78(5):1460 – 1480, 2012.
[11] M. J. Johnson and A. S. Willsky. Bayesian nonparametric hidden semi-markov models. Journal of Machine Learning Research, 14:673–701, February 2013.
[12] H. A. Kiers. Towards a standardized notation and terminology in multiway analysis. Journal of chemometrics, 14(3):105–122, 2000.
[13] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. SIAM Review, 51(3):455–500, 2009.
[14] B. Matthews, S. Das, K. Bhaduri, K. Das, R. Martin, and N. Oza. Discovering anomalous aviation safety events using scalable data mining algorithms. Journal of Aerospace Information Systems, 10(10):467–475, 2013.
A Appendix: Initial and Final Parts of HSMM

In this section we present the derivations for the initial and final steps of HSMM, which were omitted from the main text. Specifically, this amounts to computing the factor $X$ for two parts of the model, corresponding to $X_{\text{root}}$ and $X_T$ in Figures 10 and 11. The derivations for all other parts of HSMM were presented in the main text and this supplement.

Figure 10: Part of HSMM corresponding to the initial time stamps and the related part of junction tree.
To begin, recall the expression for the joint likelihood of the observed sequence:

$$P_{o_1, ..., o_T} = \prod_i d_{i-1} | x_{i-1}, x_i, \Omega \times x_{i-1} d_{i-1} \left( \begin{array}{c} x_i | x_{i-1}, d_{i-1} \end{array} \right) \times x_i o_{i | x_i}$$

and rewrite the above expression by keeping only the initial and final factors:

$$P_{o_1, ..., o_T} = \left( \Omega \times x_1 \left( \begin{array}{c} x_{2:2} \times x_2 o_{2 | x_2} \end{array} \right) \right) \times x_2 d_{1:2} \times d_{1:2} \times \cdots$$

$$\cdots \times d_{T-1:1} | x_{T-1}, d_{T-1} \times x_{T-1} d_{T-1} \left( \begin{array}{c} x_T | x_{T-1}, d_{T-1} \end{array} \right) \times x_T o_{T | x_T} \right).$$

(45)

Introduce the identity tensors into (45), regroup the terms and extract the factors $X$:

$$\tilde{X}_{o_1} = \mathcal{F} \times x_1 \left( \begin{array}{c} x_{2:2} \times x_2 \mathcal{F} \end{array} \right) \times x_2 d_{1:2} \times \mathcal{F}_{o_1}$$

(46)

and observe that the last factor $\mathcal{F}$ in (49) is a conditional probability distribution, which has the following marginalization property

$$\mathcal{F} \times x_{d_{1-1}} | x_{d_{1-2}} \frac{1}{x_{d_{1-1}}} = 1,$$

(50)

where $\mathbf{1}$ is the tensor, which has all elements equal to 1. The above can also be written in the scalar notations, $\sum_{x_{d_{1-1}}} p(x_{d_{1-1}}) = 1$ for each value of $x_{d_{1-1}}$. Therefore, if we apply (50) to (49), we get $\tilde{X}$, which is the time-shifted version of $X_{T-1}$. Therefore, to compute (47), we estimate the tensor in (12), i.e.,

$$o_{o_{T-1}} \tilde{X}_{o_{T-1}} = M^{-1} \times o_{o_{T-1}} M_{o_{T-1}},$$

(48)

Figure 11: Part of HSMM corresponding to the final time stamps and the related part of junction tree.
and marginalize out the right set of modes, corresponding to $O_{R_t}$. Alternatively, we can use the batch estimate

$$\hat{\mathbf{x}} = \left( \sum_t \alpha_{L_t} \beta_{R_t} \right)^{-1} \times \alpha_{L_t} \left( \sum_t \alpha_{L_t} \frac{M}{O_{R_t}} \right),$$

and similarly perform the marginalization. This concludes our derivations.