2010 IEEE International Workshop on Machine Learning for Signal Processing (MLSP 2010)
August 29 – September 1, 2010, Kittilä, Finland

NON-NEGATIVE MATRIX FACTORIZATION FOR PARAMETER ESTIMATION IN HIDDEN MARKOV MODELS

Balaji Lakshminarayanan and Raviv Raich

School of EECS, Oregon State University, Corvallis, OR 97331-5501
{lakshmba,raich@eecs.oregonstate.edu}

ABSTRACT

Hidden Markov models are well-known in analysis of random processes, which exhibit temporal or spatial structure and have been successfully applied to a wide variety of applications such as but not limited to speech recognition, musical scores, handwriting, and bio-informatics. We present a novel algorithm for estimating the parameters of a hidden Markov model through the application of a non-negative matrix factorization to the joint probability distribution of two consecutive observations. We start with the discrete observation model and extend the results to the continuous observation model through a non-parametric approach of kernel density estimation. For both the cases, we present results on a toy example and compare the performance with the Baum-Welch algorithm.

1. INTRODUCTION

Hidden Markov models (HMMs) are now well-known for over four decades. HMMs are used to characterize processes with specific temporal or spatial structure and provide significant advantage over models that assume independence, when the temporal or spatial independence assumption is invalid. For example in processing text documents, words exhibit dependence on the preceding content and the use of HMM may lead to improved modeling and inference as compared to an independent word model. HMM is deployed in a variety of applications such as but not limited to speech processing, motion capture, bio-informatics, and hand-writing (see [1] for a review of the topic and related references therein). In classification, a HMM can be extracted for each class (e.g., using the maximum likelihood (ML) principle) in the training phase and the Bayes risk minimizing maximum a-posteriori classifier can be used to optimally determine which of several HMMs is most likely to generate a sequence of test observations.

Commonly used in implementing HMM ML parameter estimation is the well-known Baum-Welch algorithm [1]. The Baum-Welch algorithm is a generalized expectation-maximization (EM) implementation of the ML estimator. Generalized EM is guaranteed to produce a sequence of non-decreasing likelihood values converging to a local maximum of the likelihood and as such Baum-Welch inherits such property. On the other hand, fast convergence is not guaranteed. With the Baum-Welch algorithm, the computational complexity of each iteration depends linearly on the length of the observation sequence. When dealing with long sequences, the use of Baum-Welch algorithm may be prohibitive if a lot of iterations are required. We are interested in developing methods that alleviate this problem.

Non-negative matrix factorization (NNMF) [2] is a computationally efficient method for factorizing a matrix into non-negative factors and has been successfully applied for various applications such as text mining, spectral data analysis [3], face recognition [4] etc. There has been little previous work in the application of NNMF methods for HMM. In [5], the authors employ a clever reparametrization of the traditional HMM problem [1] and compute a global solution based on singular value decomposition (SVD) for their proposed parametrization. In [6], the authors employ NNMF on higher-order transition matrices (unlike the first-order transition matrix used in this paper) to compute the parameters of HMM. However, their NNMF formulation is based on minimization of I-divergence unlike the least squares criterion used in this paper. Moreover, both of these papers assume that the observations are discrete. To the best of our knowledge, there has been no previous work on the application of NNMF to HMMs with continuous observations.

In this paper, we present a novel estimator of the HMM parameters that is based on a rank-constrained NNMF of the empirical estimate of the joint distribution of two consecutive observations. For observations that are discrete random variables, we present a solution that is directly based on NNMF. The computational complexity of our method is independent of the length of the data sequence. We extend the method for observations that are continuous random variables through a kernel formulation. We present two toy examples to illustrate the performance of the method. For the discrete HMM, we provide both qualitative and quantitative comparisons of our method to the Baum-Welch algorithm. For the continuous observations case, we provide a qualitative comparison to the Baum-Welch algorithm.

In Section 2, we formulate the problem of HMM parameter estimation for the discrete and continuous observations model. In Section 3, we describe the algorithms to estimate the HMM parameters. In Section 4, we present two toy examples to demonstrate the performance of the proposed algorithms. Finally, Section 5 provides a summary of the paper.

2. PROBLEM FORMULATION

We consider the HMM as presented through the graphical model in Fig. 1(a). The hidden state at time i is denoted by $s(i)$ and the observed output at time i is denoted by $x(i)$. We assume that the state is a discrete random variable (RV), which takes upon one of L values from $\mathcal{S} = \{1, 2, \ldots, L\}$. The state transition probability is denoted by $S(s_2|s_1) \triangleq P(s(i+1) = s_2|s(i) = s_1)$. We make the assumption that the process is stationary and hence we can write the stationary state probability $S(s) \triangleq P(s(i) = s)$. We choose the state joint probability $S(s_1, s_2) = P(s(i) = s_1, s(i+1) = s_2)$ instead of the transition matrix to represent the model. Note that the joint state probability $S(s_1, s_2)$ can be obtained from the

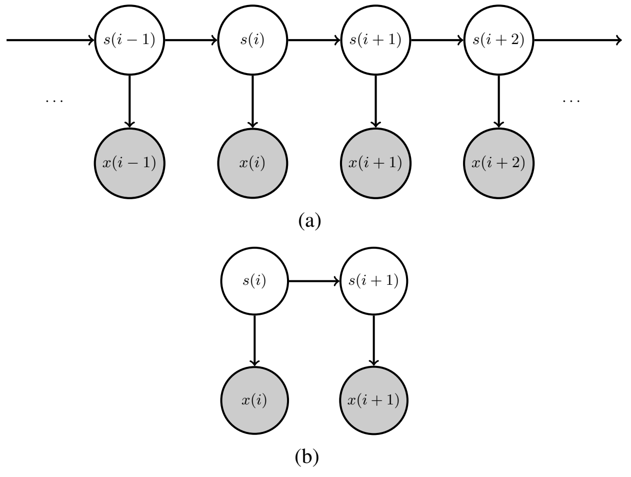


Fig. 1. Graphical models for HMM (a) a complete Markov chain (b) a two-step Markov chain

transition $S(s_2|s_1)$ and the stationary state probability $S(s_1)$ by Bayes rule $S(s_1, s_2) = S(s_2|s_1)S(s_1)$. Moreover, the stationary marginal $S(s_1)$ can be obtained from the transition probability by solving the linear equation $S(s_2) = \sum_{s_1} S(s_2|s_1)S(s_1)$ for all $s_2 \in \mathcal{S}$. The observation $x(i)$ can be either discrete or continuous and takes upon a value in $\mathcal{X}$. We denote the conditional distribution of the observation $x(i)$ given the state $s(i)$ by $P(x|s) \triangleq P(x(i) = x|s(i) = s) : \mathcal{X} \times \mathcal{S} \to \mathbb{R}$. Given $\mathcal{S}$ and $\mathcal{X}$, the process can be parameterized by the pair $\lambda = \{S(\cdot,\cdot), P(\cdot|\cdot)\}$. We refer the reader to [1] for a more detailed introduction to HMM.

The core of the HMM parameter estimation method presented here relies on the factorization of the joint probability of two consecutive observations $Q(x_1, x_2) \triangleq P(x(i+1) = x_2, x(i) = x_1)$. Note that we are estimating the parameters of HMM based on the model in Fig. 1(b). We can write $Q(x_1, x_2)$ as

$$Q(x_1, x_2) = \sum_{s_1, s_2} P(x_1|s_1)S(s_1, s_2)P(x_2|s_2). \tag{1}$$

If $x(i)$ is a discrete RV and $\mathcal{X} = \{1, 2, \ldots, M\}$, then (1) can be written in a matrix form as

$$Q = PSP^T. \tag{2}$$

where Q, P, and S are $M \times M$, $M \times L$ and $L \times L$ matrices, respectively. Note that in this matrix form, all three matrices Q, P, and S are non-negative and that the sum-to-one property for Q and S is of the form $\sum_{x_1,x_2} Q(x_1, x_2) = 1$ and $\sum_{s_1,s_2} S(s_1, s_2) = 1$ or in matrix notation $1^T Q1 = 1$ and $1^T S1 = 1$, where 1 is the vector of all ones. The sum-to-one property for P is of the form $\sum_{x_1} P(x_1|s_1) = 1$ for all $s_1 \in \mathcal{S}$ or in matrix notation $1^T P = 1^T$.

2.1. Discrete observations HMM parameter estimation

In general, the problem of HMM parameter estimation can be stated as: find P and S given the observations: $x(1), x(2), \ldots, x(n)$. Here, we restrict the problem to finding P and S using the empirical estimate of Q given by

$$\hat{Q}(x_1, x_2) = \frac{1}{n-1}\sum_{i=1}^{n-1} I(x(i) = x_1)I(x(i+1) = x_2), \tag{3}$$

for all $(x_1, x_2) \in \mathcal{X} \times \mathcal{X}$. We would like to point out that the simplicity of our formulation (in contrast to [6]) comes with a cost. Not all necessary statistics are used and hence the efficiency of an estimator that relies only on $\hat{Q}$ is not guaranteed. We consider the nonlinear least-squares criterion for estimating P and S based on the following optimization problem:

$$\begin{aligned} \min_{\substack{P\in\mathbb{R}^{M\times L}\\ S\in\mathbb{R}^{L\times L}}} \quad & \|\hat{Q} - PSP^T\|^2 \\ \text{subject to} \quad & 1^T P = 1^T,\ 1^T S1 = 1,\ P \geq 0,\ S \geq 0, \end{aligned} \tag{4}$$

where $X \geq 0$ is used here as a shorthand notation for $X_{ij} \geq 0$ for all i, j. Note that an alternative formulation of the problem using the KL-divergence can be considered as well:

$$\begin{aligned} \min_{\substack{P\in\mathbb{R}^{M\times L}\\ S\in\mathbb{R}^{L\times L}}} \quad & \sum_{ij} \hat{Q}_{ij} \log(\hat{Q}_{ij}/(PSP^T)_{ij}) \\ \text{subject to} \quad & 1^T P = 1^T,\ 1^T S1 = 1,\ P \geq 0,\ S \geq 0. \end{aligned} \tag{5}$$

In this paper, we proceed concentrating only on the problem formulation in (4). Note that this problem is not convex and may have multiple local minima. Key to the algorithm that is presented in the next section is the low-rank factorization of the $M \times M$ matrix Q into a product of $M \times L$, $L \times L$, and $L \times M$ matrices. Note that except for the creation of $\hat{Q}$ in (3), the problem described in (4) does not depend on n, the length of the observation sequence. Hence, this problem formulation is attractive when dealing with long sequences.

2.2. Extension to continuous observations

Next, we consider the continuous observations model. The notations vary in the following:

1. $\mathcal{X} = \mathbb{R}^d$
2. $p(x(i) = x|s(i) = s)$ corresponds to the probability density function of the observations, i.e., $\int_{\mathcal{X}} p(x(i) = x|s(i) = s)dx = 1$ and $p(x(i) = x|s(i) = s) \geq 0$.
3. $p(x(i) = x_1, x(i+1) = x_2)$ is the joint probability density function of $x(i)$ and $x(i+1)$.

With these modifications, the low rank decomposition in (1) remains exact. To obtain information about $p(x|s_1)$ for all $x \in \mathcal{X}$, a function version of (2) is required. However, to perform a factorization similar to (4), a matrix formulation is required. First, consider a finite sample set $\tilde{\mathcal{X}} = [x^{(1)}, x^{(2)}, \ldots, x^{(m)}]$, which in some sense covers $\mathcal{X}$. This can be achieved by increasing m to reach a desirable accuracy of covering $\mathcal{X}$. Using this approach, the LHS of (1) can be evaluated over the grid $\tilde{\mathcal{X}} \times \tilde{\mathcal{X}}$ as

$$Q(x^{(i)}, x^{(j)}) = \sum_{s_1,s_2} p(x^{(i)}|s_1)S(s_1, s_2)p(x^{(j)}|s_2), \tag{6}$$

to produce the desired matrix formulation. We are interested in a solution that will provide $p(x|s)$ for all $x \in \mathcal{X}$. This requires an interpolation of $p(x|s)$ to values other than $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$.

To this end, we consider reformulating the problem using a kernel density estimate of $p(x|s)$:

$$p(x|s) = \sum_{k=1}^{n} \alpha_{ks} K(x, x_k), \tag{7}$$

where α_{ks} are non-negative coefficients $\alpha_{ks} \geq 0$ that sum to one $\sum_k \alpha_{ks} = 1$ for all s and $K(x,y)$ is the kernel of the density estimator. A wide variety of kernels can be deployed [7]. We restrict the kernel to satisfy the following properties

1. Non-negativity $K(x,y) \geq 0$,
2. Symmetry $K(x,y) = K(y,x)$,
3. Sum-to-one $\int_{\mathcal{X}} K(x,y)dx = 1$,
4. First order moment $\int_{\mathcal{X}} xK(x,y)dx = y$, and
5. Bandwidth $\int_{\mathcal{X}} \|x-y\|^2 K(x,y)dxdy = \sigma^2 d$,

where d is the dimensionality of the observations. Bandwidth selection can be made to balance bias and variance in the density estimator as in [8]. Next, we consider the kernel density estimator of $Q(x,x')$ of the form

$$\hat{Q}(x,x') = \frac{1}{n-1} \sum_{k=1}^{n-1} K(x,x_k)K(x',x_{k+1}). \tag{8}$$

Using the kernel estimate of Q in (8) and the kernel representation of $p(x|s)$ as in (7), we introduce a kernelized version of (6):

$$\frac{1}{n-1} \sum_{k=1}^{n-1} K(x^{(i)},x_k)K(x^{(j)},x_{k+1}) =$$

$$\sum_{s_1,s_2} \Big(\sum_{k_1=1}^{n} \alpha_{k_1 s_1} K(x^{(i)},x_{k_1}) S(s_1,s_2) \sum_{k_2=1}^{n} \alpha_{k_2 s_2} K(x^{(j)},x_{k_2}) \Big).$$

The matrix notation of the above equation is

$$\hat{Q} = \frac{1}{n-1} KDK^T = KASA^TK^T, \tag{9}$$

where $K_{ik} = K(x^{(i)},x_k)$, $D_{kl} = \delta_{k+1,l}$, and $A_{ks} = \alpha_{ks}$. Here, $\hat{Q}$, K, A and S are $m \times m$, $m \times n$, $n \times L$ and $L \times L$ matrices, respectively. Note that (9) is a set of m^2 equations with a number of parameters $nL + L^2$ and a number of constraints $L+1$. A necessary condition for a unique solution is that $m^2 \geq nL + L^2 - L - 1$. For a large number of samples n, we can choose m to be of the order of $\sqrt{nL}$ to satisfy this inequality. Choosing $\tilde{\mathcal{X}} = \mathcal{X}$ yields n^2 equations, which leads to a ratio of equations to parameters of the order of $\frac{n}{L}$. For large n, this ratio could potentially lead to a more accurate estimation. Alternatively, a computationally efficient choice of $\tilde{\mathcal{X}}$ can be made by sub-sampling $\mathcal{X}$ so that m is of the order of $\sqrt{nL}$ thus leading to a number of equations, which is linear in the number of points n trading-off estimation accuracy with computational complexity.

We choose the nonlinear least-squares criterion for estimating A and S based on the following optimization problem:

$$\min_{\substack{A \in \mathbb{R}^{n\times L} \\ S \in \mathbb{R}^{L \times L}}} \quad \|\hat{Q} - KASA^TK^T\|^2$$

$$\text{subject to} \quad 1^TA = 1^T,\ 1^TS1 = 1,\ A \geq 0,\ S \geq 0. \tag{10}$$

In the discrete case, the problem in (4) did not depend on n. However, in the continuous case, the problem described in (10) implicitly depends on n through m. The choice of m presents a trade-off between computational complexity and accuracy of estimation.

3. ALGORITHMS

In this section, we present algorithms to solve the optimization problems described in (4) and (10). To solve the problem in (4), we propose an approach motivated by NNMF. NNMF is a well studied problem and is a popular approach when the factor representations need to be non-negative. Two popular algorithms for NNMF are alternating least squares and multiplicative update algorithm. We refer the reader to [3] for more details regarding NNMF. For a discussion about the convergence properties of alternating least squares, we refer to [9]. Note that the problem in (4) is not a straightforward NNMF since the optimization function involves fourth order terms in P.

Following the approach of alternating non negative least squares for NNMF, we propose the following iterative algorithm. We initialize $\hat{P}$ to lie in the column space of $\hat{Q}$. We find the least squares solution for S given by $S_{LS} = \hat{P}^\dagger \hat{Q} \hat{P}^{\dagger T}$ where $\hat{P}^\dagger$ denotes the pseudo-inverse of $\hat{P}$. Next, we project S_{LS} onto the positive quadrant, $S_1 = S_{LS}^+ = \max(S_{LS}, 0)$ and then impose the sum-to-one constraint using $\hat{S} = S_1/(1^TS_11)$. Next, we update $\hat{P}$ by solving for $\hat{Q} = \hat{P}_{old}SP^T$ where $\hat{P}_{old}$ denotes the previous value of $\hat{P}$ and is treated as a constant. We find the least squares solution for P and then project it onto the positive quadrant i.e., $P_1 = \max(((\hat{P}\hat{S})^\dagger \hat{Q})^T, 0)$. Next, we update $\hat{P}$ by normalizing each column of P_1. Next, we update $\hat{S}$ again using the new $\hat{P}$. Finally, we update $\hat{P}$ again by solving for $\hat{Q} = PS\hat{P}_{old}^T$ using a similar approach to that used for solving $\hat{Q} = \hat{P}_{old}SP^T$. The complete pseudo code for the discrete version of the NNMF-HMM algorithm is shown in Algorithm 1. A complete proof of convergence for Algorithm 1 is beyond the scope of this paper.

Algorithm 1 NNMF-HMM Discrete case

```
Construct matrix $\hat{Q}$ by (3)
Initialize $\hat{P}$ to be in column space of $\hat{Q}$
**repeat**
    $S_1 = \max(\hat{P}^\dagger \hat{Q} \hat{P}^{\dagger T}, 0)$
    Normalize: $\hat{S} = S_1/(1^TS_11)$
    $P_1 = \max(((\hat{P}\hat{S})^\dagger \hat{Q})^T, 0)$
    Normalize: $\hat{P}_{ij} = P_{1ij}/(\sum_i P_{1ij})$
    $S_1 = \max(\hat{P}^\dagger \hat{Q} \hat{P}^{\dagger T}, 0)$
    Normalize: $\hat{S} = S_1/(1^TS_11)$
    $P_2 = \max(((\hat{P}\hat{S}^T)^\dagger \hat{Q}^T)^T, 0)$
    Normalize: $\hat{P}_{ij} = P_{2ij}/(\sum_i P_{2ij})$
**until** $\|\hat{Q} - \hat{P}\hat{S}\hat{P}^T\|^2$ has converged
```

To solve the problem in (10), we use the following algorithm. We first compute the $m \times n$ matrix K by evaluating the value of kernel at m input points. As with the discrete case, we propose an alternating least squares approach following the same cycle of estimation. Here K is incorporated in the least squares (LS) estimation of A and S. We use non-negative least squares (NNLS) procedure for solving A and S. The pseudo code for the continuous version of the NNMF-HMM algorithm is shown in Algorithm 2.

4. NUMERICAL RESULTS

In this section, we present a toy example for each of the scenarios, i.e., discrete observations and continuous observations.

Algorithm 2 NNMF-HMM Continuous case

```
Estimate kernel bandwidth $\sigma$.
Compute $m = \lceil\sqrt{nL + L^2}\rceil$ and evaluate the kernel matrix $K$ at $m$ input points
Construct the $m \times m$ matrix $\hat{Q}$ according to (8)
Initialize $\hat{A}$
repeat
    Solve using NNLS: $S_1 = \min_{S>0} \|\hat{Q} - K\hat{A}S\hat{A}^T K^T\|^2$
    Normalize: $\hat{S} = S_1/(1^T S_1 1)$
    Solve using NNLS: $A_1 = \min_{A>0} \|\hat{Q} - KAS\hat{A}_{old}^T K^T\|^2$
    Normalize: $\hat{A}_{ij} = A_{1ij}/(\sum_i A_{1ij})$
    Solve using NNLS: $S_1 = \min_{S>0} \|\hat{Q} - K\hat{A}S\hat{A}^T K^T\|^2$
    Normalize: $\hat{S} = S_1/(1^T S_1 1)$
    Solve using NNLS: $A_2 = \min_{A>0} \|\hat{Q} - K\hat{A}_{old}SA^T K^T\|^2$
    Normalize: $\hat{A}_{ij} = A_{2ij}/(\sum_i A_{2ij})$
until $\|\hat{Q} - K\hat{A}\hat{S}\hat{A}^T K\|^2$ has converged
```

4.1. Discrete Observations

We tested the performance of the proposed algorithm using a toy example. We considered a HMM with $L = 3$ states. We used the following transition matrix

$$[S(s_2|s_1)] = \begin{bmatrix} 0 & 0.9 & 0.1 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{11}$$

to produce $s(i+1) = s_2$ given $s(i) = s_1$. To produce the observations, we first created x' according to the conditional model given by $x'|s = 1 \sim \mathcal{N}(11, 2)$, $x'|s = 2 \sim \mathcal{N}(16, 3)$, $x'|s = 3 \sim U[16, 26]$. Then, we generated x by rounding x' to the nearest integer. Following this model, we generated the data for different values of n, $n = 10^3$, $n = 10^4$ and $n = 10^5$ observations. The joint probability of two consecutive observations $Q(x_1, x_2)$ is then formed by (3) and Algorithm 1 described in Section 3 was applied to estimate $P(x|s)$ and $S(s_1, s_2)$. The convergence criterion used was that the absolute difference between the current value of the error (given by $\|\hat{Q} - \hat{P}\hat{S}\hat{P}^T\|^2$) and the previous value of error should be less than 10^{-5}. Since the NNMF-HMM algorithm converges to local optimum, we repeat the procedure with 5 random initializations and use the solution with the highest log likelihood. For each value of n (the number of samples), we generate the data 10 times and obtain 10 independent estimates of $P(x|s)$ presented in Figures 2(a), 2(b) and 2(c). We observe that

- Despite the overlap between the supports of the probability models for the observations, the estimator was able to extract $P(x|s)$ in areas of overlap.
- Despite the non-Gaussian nature of $P(x|s = 3)$, it was estimated consistently.
- As n increases, the estimates concentrate around the true value of $P(x|s)$.

4.2. Comparison to Baum-Welch algorithm - Discrete case

In this section, we present both qualitative and quantitative comparisons of the proposed approach to the Baum Welch algorithm. For qualitative comparison, we provide the $\hat{P}(x|s)$ estimates obtained by the two methods. For quantitative evaluation, we compare the log likelihood, Hellinger distance of $\hat{P}(x|s)$ to the true $P(x|s)$ and the runtime of NNMF and HMM for different values of the number of samples n. For details regarding the Baum-Welch algorithm, we refer the reader to [1]. We used the Matlab implementation of the Baum Welch algorithm available in Matlab-HMM toolbox [10] for all of our simulations.

The NNMF-HMM procedure is the same as that described in Section 4.1. Similar to NNMF, we repeat the Baum-Welch procedure with 5 random initializations and pick the solution with the highest log likelihood. We set the '`thresh`' criterion in the HMM toolbox as 10^{-5} and limit the maximum number of Baum Welch iterations to 500. Similar to Section 4.1, we obtain 10 different estimates for each value of n. Comparing Figures 2(a), 2(b) and 2(c) to Figures 2(d), 2(e) and 2(f), we observe that the estimates of $P(x|s)$ obtained by NNMF-HMM procedure are comparable to the estimates obtained using Baum-Welch algorithm.

Next, we provide quantitative comparisons. First, we compare the total run time (includes the time for each of the 5 random initializations) of Baum-Welch and NNMF-based approach as a function of n, the number of samples. For each fixed value of n, we generate 10 independent realizations of the data and compute the mean of the 10 values of the total runtime. We also compare the log-likelihood and the Hellinger distance to the true $P(x|s)$ for the two methods. Similar to the runtime, we report the mean values of log likelihood and Hellinger distance across 10 independent datasets for each value of n. The total Hellinger distance was computed as $d_H^{total} = \sum_{i=1}^{L} d_H(P(x|s = i), \hat{P}(x|s = i))$, where

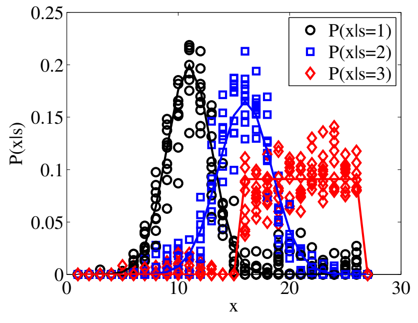
$$d_H^2(p, q) = \frac{1}{2}\sum_{j=1}^{M}(\sqrt{p_j} - \sqrt{q_j})^2.$$

The results are shown in Fig. 3. We observe that the NNMF-HMM is about two orders of magnitude faster than Baum-Welch algorithm, while providing comparable values of log-likelihood. The Hellinger distance of NNMF-HMM procedure is higher compared to Baum-Welch since NNMF-HMM optimizes least squares criterion instead of maximum likelihood. We note that alternative formulations of NNMF-HMM such as (5) can be potentially used to obtain lower values of Hellinger distances.

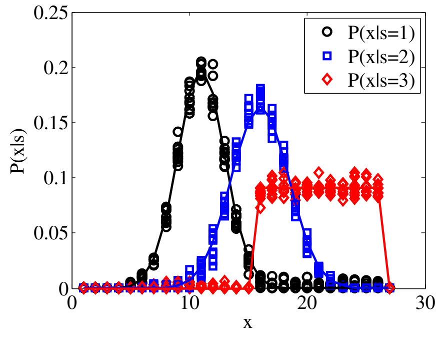
The complexity of Baum-Welch algorithm is $O(I_1 L^2 n)$ where I_1 is the number of iterations required for the Baum-Welch procedure to converge. The computational complexity of each iteration of the Baum-Welch algorithm is quadratic in L, the number of states, and linear in n, the length of the data sequence. In the NNMF-HMM procedure, the creation of $\hat{Q}$ matrix involves an $O(n)$ operation. Each iteration of NNMF-HMM algorithm requires inversion of $M \times L$ matrix, which takes $O(M^2 L + L^3)$. Usually $L \ll M$ and hence, the total complexity of our method is $O(n + I_2(M^2 L))$ where I_2 is the number of iterations required by the NNMF-HMM procedure. If $n \gg M^2 L$, the complexity of our method is just $O(n)$, whereas the Baum-Welch procedure has complexity $O(I_1 L^2 n)$, which indicates that our method is roughly $O(I_1 L^2)$ times faster than Baum-Welch. If $n \gg M$, our method is more than an order of magnitude faster than the Baum-Welch procedure as shown in Fig. 3(a). We note that the NNMF-HMM procedure can also be used as a good initialization procedure to reduce I_1, the number of Baum-Welch iterations.

4.3. Continuous observations

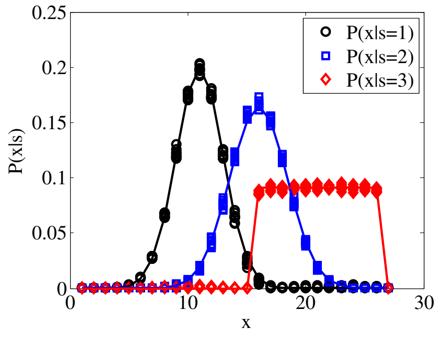
To evaluate the continuous observations estimator in Section 2.2, we generated data using the same setup as in Section 4.1, but omit-



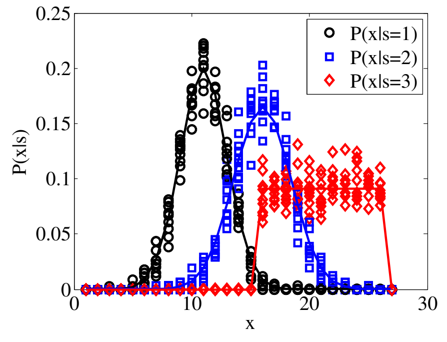
(a) $n = 10^3$ (NNMF-HMM)



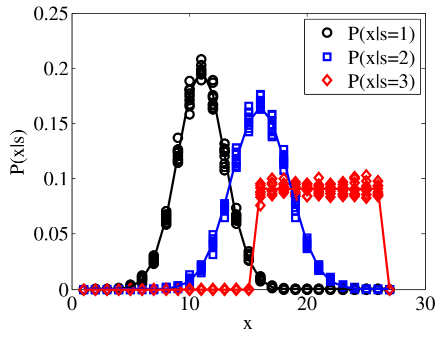
(b) $n = 10^4$ (NNMF-HMM)



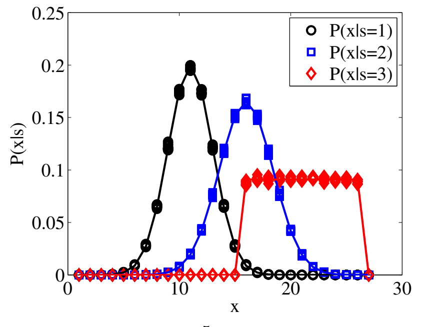
(c) $n = 10^5$ (NNMF-HMM)



(d) $n = 10^3$ (Baum-Welch)

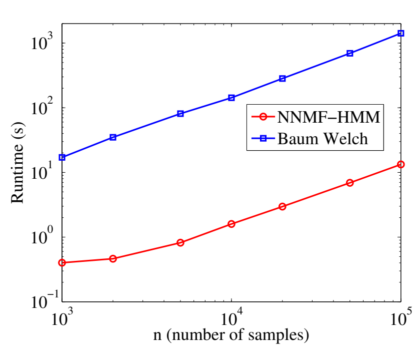


(e) $n = 10^4$ (Baum-Welch)

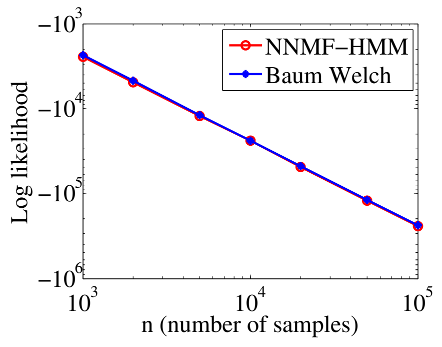


(f) $n = 10^5$ (Baum-Welch)

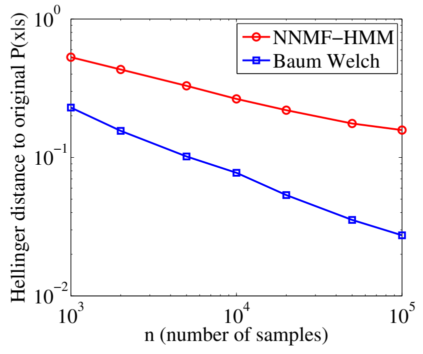
Fig. 2. Comparison of NNMF-HMM and Baum Welch - Discrete case. Each graph contains 10 different estimates of $P(x|s)$ along with the true $P(x|s)$ distribution.



(a)

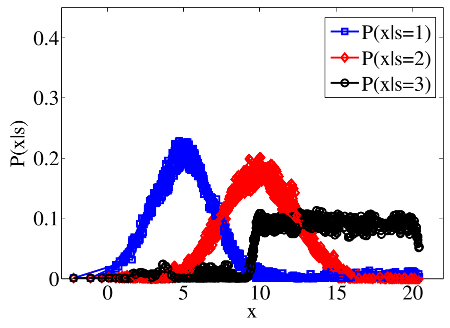


(b)

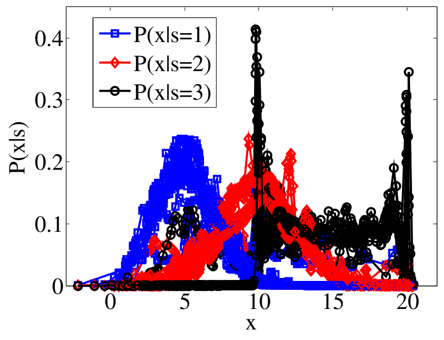


(c)

Fig. 3. Comparison of NNMF-HMM to Baum-Welch algorithm for the discrete observations case a) Run time, b) Log-likelihood and c) Hellinger distance to original $P(x|s)$.



(a) NNMF-HMM



(b) Baum Welch for HMM-GMM

Fig. 4. Comparison of NNMF-HMM to Baum-Welch algorithm for the continuous case: $\hat{p}(x|s)$ for $n = 5000$. Each figure contains 10 different estimates of $p(x|s)$.

ted the rounding of x' to the nearest integer. We considered the Gaussian kernel and used the ML bandwidth estimator [8]. We used $n = 5000$ and $m = \lceil\sqrt{nL + L^2}\rceil = 123$. For obtaining the NNLS solution in Algorithm 2, we used projected Landweber iterations. As before, we repeated the procedure 10 times to produce 10 independent estimates of $p(x|s)$, which are presented in Fig. 4(a). We observe that, as in the discrete case, the estimator was able to extract $p(x|s)$ even in areas where the distributions overlap.

4.4. Comparison to Baum-Welch algorithm - Continuous case

To compare the performance of NNMF-HMM to Baum-Welch for the continuous case, we consider the HMM with mixture of Gaussians outputs in the HMM toolbox [10] i.e. we fit a GMM for each of the $p(x|s)$. Due to the computationally intensive nature of the continuous case, we just present qualitative results for one of the settings. We generated data as described in Section. 4.3. We fixed the number of mixture components to be $m = 123$. Since the Baum-Welch algorithm can only approximate the solution (using a Gaussian mixture to approximate the uniform distribution), we allow it to use a comparable number of mixture components as with the NNMF-HMM algorithm. As before, we generate the data 10 times and obtain 10 different estimates of $p(x|s)$. The 10 estimates for $P(x|s = i)$ with $i = 1, 2, 3$ for the NNMF-HMM algorithm are presented in Fig. 4(a) and for the Baum Welch algorithm in Fig. 4(b). Comparing the two figures in Fig. 4, we can observe that the estimates produced by NNMF-HMM follow the original distributions more accurately, especially for the uniform distribution.

5. CONCLUSION

We presented a novel algorithm for estimating the parameters of an HMM through the application of NNMF to the joint probability distribution of two consecutive observations. Specifically, we presented a solution for the discrete observation model and extended the results to continuous observation through the non-parametric approach using kernel density estimation. For the discrete case, we showed that our algorithm can be significantly faster than the Baum-Welch algorithm while providing comparable values of log-likelihood. For the continuous case, we showed that our algorithm can produce more accurate solutions for non-Gaussian densities. The simplicity and computational efficiency of the algorithm makes it attractive both as an alternative to existing algorithms (such as the Baum-Welch algorithm) and as a possible initialization method to more accurate but cumbersome methods. Future work will explore the application of these algorithms to real world datasets.

6. REFERENCES

[1] L.R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[2] D.D. Lee and H.S. Seung, "Algorithms for non-negative matrix factorization," *Advances in neural information processing systems*, vol. 13, 2001.

[3] M.W. Berry, M. Browne, A.N. Langville, V.P. Pauca, and R.J. Plemmons, "Algorithms and applications for approximate nonnegative matrix factorization," *Computational Statistics & Data Analysis*, vol. 52, no. 1, pp. 155–173, 2007.

[4] D.D. Lee and H.S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[5] D. Hsu, S.M. Kakade, and T. Zhang, "A spectral algorithm for learning hidden markov models," *Proceedings of the Conference on Learning Theory (COLT)*, 2009.

[6] G. Cybenko and V. Crespi, "Learning Hidden Markov Models using Non-Negative Matrix Factorization," *Arxiv preprint arXiv:0809.4086*, 2008.

[7] D.W. Scott, *Multivariate density estimation: theory, practice, and visualization*, Wiley-Interscience, 1992.

[8] B.A. Turlach, "Bandwidth selection in kernel density estimation: A review," *CORE and Institut de Statistique*, pp. 23–493, 1993.

[9] C.J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Computation*, vol. 19, no. 10, pp. 2756–2779, 2007.

[10] K. Murphy, "Hidden markov model (hmm) toolbox for matlab," *online at http://www.ai.mit.edu/˜murphyk/Software/HMM/hmm.html*.