# Orthogonal Mixture of Hidden Markov Models

Negar Safinianaini[1(✉)], Camila P. E. de Souza[2], Henrik Boström[1], and Jens Lagergren[1,3]

[1] School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, Stockholm, Sweden
{negars,bostromh}@kth.se
[2] Department of Statistical and Actuarial Sciences, The University of Western Ontario, London, Canada
camila.souza@uwo.ca
[3] Science for Life Laboratory, Solna, Sweden
jens.lagergren@scilifelab.se

**Abstract.** Mixtures of Hidden Markov Models (MHMM) are widely used for clustering of sequential data, by letting each cluster correspond to a Hidden Markov Model (HMM). Expectation Maximization (EM) is the standard approach for learning the parameters of an MHMM. However, due to the non-convexity of the objective function, EM can converge to poor local optima. To tackle this problem, we propose a novel method, the Orthogonal Mixture of Hidden Markov Models (oMHMM), which aims to direct the search away from candidate solutions that include very similar HMMs, since those do not fully exploit the power of the mixture model. The directed search is achieved by including a penalty in the objective function that favors higher orthogonality between the transition matrices of the HMMs. Experimental results on both simulated and real-world datasets show that the oMHMM consistently finds equally good or better local optima than the standard EM for an MHMM; for some datasets, the clustering performance is significantly improved by our novel oMHMM (up to 55 percentage points w.r.t. the v-measure). Moreover, the oMHMM may also decrease the computational cost substantially, reducing the number of iterations down to a fifth of those required by MHMM using standard EM.

**Keywords:** Hidden markov models · Mixture models · Mixture of hidden markov models · Expectation maximization · Orthogonality · Regularization · Penalty

## 1 Introduction

Clustering of sequential data is an important machine learning task with a wide range of applications from biology to finance. Various methods have been applied for this task, including feature-based methods, deep-learning approaches and model-based methods [1,6,19]. Model-based methods have the advantages of


F. Hutter et al. (Eds.): ECML PKDD 2020, LNAI 12457, pp. 509–525, 2021.
https://doi.org/10.1007/978-3-030-67658-2_29

being probabilistic, interpretable, providing measures of uncertainty, as well as allowing for rapid prototyping [7,33]. Motivated by these advantages, we focus on model-based approaches; more specifically on the Hidden Markov Model (HMM), or a Mixture of Hidden Markov Models (MHMM), introduced in [29]. They have been applied to tasks such as time series clustering, music analysis, motion recognition, handwritten digit recognition, as well as to problems within finance and biology [9,12,13,16,23,30,34]. In recent years, various methods have been proposed to enhance an MHMM with respect to, e.g., time complexity and sparsity of the model [24,30], when using Expectation Maximization (EM) [10], which is the standard inference approach when clustering based on an MHMM [6]. However, since EM maximizes a non-convex objective function, it can lead to poor local optima; a proper initialization plays a crucial role here [6,8]. Moreover, singularity in mixture model estimation is also a potential problem; i.e., a cluster can be collapsed into a single data point [6]. To tackle the first problem, it is common to consider several random initializations [6,8]. For the second problem, the inclusion of a penalty or prior on the optimization variable has been suggested as a solution [6]. In general, to enhance the predictive power of a model, an HMM in this context, penalization can be applied on EM [20]; controlling the behavior of the model parameters. Some recently proposed penalties over HMM parameters (either transition or emission distribution parameters) include the pairwise mean difference penalty [21], the smoothing penalty [32], the sparse penalty [31], the self-transition penalty [22] and the determinantal point process [17] penalty [20].

Each of the penalties introduced in previous work is however designed for a single HMM and is consequently not particularly suited for training a mixture of HMMs. In particular, these penalties do not introduce any dependencies between the HMMs. In contrast, we introduce a novel penalty, the *orthogonality penalty*, by using the inner product as a distance measure between pairs of HMMs, or more exactly, their transition matrices. Distance measures between HMMs have been used also previously for the purpose of clustering sequences, however, without using MHMMs [15]. Specifically, the method in [15] do not use HMMs to represent clusters of sequences; instead, each sequence is represented by a unique HMM and the actual clustering is performed on the HMMs, that is, subsequent to the construction of the HMMs. When it comes to incorporating the concept of orthogonality, a similar idea was exploited in [3], where orthogonality among the weights of a neural network was used as a constraint to improve maximum likelihood estimation w.r.t. initialization sensitivity.

In this work, we modify the standard EM to infer an MHMM, where the novel orthogonality penalty is used to push the EM algorithm to avoid a poor local optimum, by increasing the distance between the transition matrices of each pair of HMMs. We term the modified EM procedure as the Orthogonal Mixture of Hidden Markov Models (oMHMM). A key feature of the oMHMM method, which is missing in an MHMM due to the independence of the transition matrices, is that the estimation of each transition matrix may be affected by all other transition matrices (representing different clusters); hence realizing

a *global context*. As we aim to improve upon the standard EM when inferring an MHMM, we compare the proposed oMHMM to the standard EM implementation of MHMM, as, to the best of our knowledge, no other approaches to infer MHMMs have been put forward in the literature[1].

In the next section, we provide notation and background on HMMs, MHMMs, and a linear algebraic definition of orthogonality. In Sect. 3, we introduce the oMHMM; our novel approach. In Sect. 4, we evaluate and compare it to standard EM for MHMM. Finally, in Sect. 5, we summarize the main findings and point out directions for future research.

## 2 Preliminaries

In this section, we first provide definitions concerning clustering of sequences using mixtures of HMMs and, then, a short description of orthogonality.

### 2.1 Hidden Markov Models

A Hidden Markov Model (HMM) is a probabilistic graphical model [5] in which a sequence of emitted symbols (observation sequence) is observed, but the sequence of states (state sequence) emitting the observation sequence, is hidden and follows a Markov structure. The Markov property implies that the next state in the state sequence only depends on the current state. We assume that the input sequences have length $M$. The observation sequence is denoted as $Y = \{y_1, \ldots, y_M\}$ and the state sequence is denoted as $C = \{c_1, \ldots, c_M\}$. Each state in $C$ takes a value $j$ for $j = 1, \ldots, J$; we denote each state at step $m$ with value $j$ as $c_{m,j}$ for $m = 1, \ldots, M$. An HMM is parameterized by initial probabilities, $p(c_{1,j})$, transition probabilities, $p(c_{m,j}|c_{m-1,i})$, and emission probabilities, $p(y_m|c_{m,j})$. These sets of parameters are referred to as $\rho$, $A$, and $O$ respectively, jointly referred to as $\theta$. Inference of HMM parameters can be conducted by maximizing the likelihood using the Expectation Maximization (EM) algorithm [6], which consists of two steps: the *E-step*, calculating $Q(\theta, \theta^{old}) = E_{C|Y,\theta^{old}}\big[\log P(Y, C|\theta)\big]$, the expected value of $\log P(Y, C|\theta)$, the *complete-data log likelihood* [6], w.r.t. the conditional distribution of the hidden states given the observed data and old parameter estimates ($\theta^{old}$); the *M-step*, maximizing the $Q$ function calculated in the $E$-step w.r.t. the parameters of interest. The $E$-step involves calculating the marginal posterior distribution of a latent variable $c_{m,j}$, denoted as $\gamma(c_{m,j})$, and the joint posterior distribution of two successive latent variables, $\varepsilon(c_{m-1,i}, c_{m,j})$. For details of the calculations of these terms, we refer to [6]. In the $M$-step, $\rho$, $O$, and $A$ are updated using $\gamma(c_{m,j})$ and $\varepsilon(c_{m-1,i}, c_{m,j})$. The $M$-step concerns a maximization problem per parameter set; here we focus on the maximization problem regarding the transition probabilities, $A$, as the contribution of this work involves this problem.

[1] Spectral learning of MHMM [30] improves the time complexity and does not improve the clustering. Sparse MHMM [24], requiring data coming from a set of entities connected in a graph with a known topology, can be used together with oMHMM.

Therefore, the maximization problem shown in Eq. 1 results from maximizing the $Q$ function w.r.t. $A_{ij}$ which is the transition probability of moving from state $i$ to state $j$; the optimization is subject to the constraint $\sum_{j=1}^{J} A_{ij} = 1$.

$$\max_{A_{ij}} \quad \sum_{m=2}^{M} \sum_{i=1}^{J} \sum_{j=1}^{J} \varepsilon(c_{m-1,i}, c_{m,j}) \log A_{ij} \tag{1}$$

### 2.2 Mixtures of Hidden Markov Models

A Mixture of Hidden Markov Models (MHMM) is a probabilistic graphical model where each observation sequence is generated by a *mixture model* [6] with $K$ components, each representing a cluster which corresponds to a unique HMM parameter setting. We denote each observation sequence as $Y_n = \{y_{n1}, \ldots, y_{nM}\}$ and $Z_n$ is the latent variable concerning the cluster assignment of $Y_n$, for $n = 1, \ldots, N$. An observation sequence $Y_n$, belonging to the $k$-th ($k$ takes a value between 1 and $K$) cluster, arises from an HMM with state sequence $C_n = \{c_1^n, \ldots, c_M^n\}$ parameterized by $\rho_k$, $O_k$, $A_k$. The parameter defining the probability of the observation sequences belonging to component $k$, the mixture probability, is $\pi_k$ with $\sum_{k=1}^{K} \pi_k = 1$. Performing EM on the MHMM, all parameters, $\rho_{1:K}$, $O_{1:K}$, $A_{1:K}$, and $\pi_{1:K}$, are updated at each iteration. Note that using these parameters, the posterior probability of each observation sequence belonging to component $k$ can be calculated; i.e. $p(Z_n = k|Y_n, \theta^{old})$. For details on the update equations, we refer to [24]. Similar to the previous section, the $E$-step concerns calculating $Q(\theta, \theta^{old}) := E_{C,Z|Y,\theta^{old}}\big[\log p(Y, C, Z|\theta)\big]$ and $M$-step maximizing the $Q$ function. Here we focus again on the $M$-step, where the maximization problem concerns the transition matrices, i.e., maximizing the $Q$ function w.r.t. $A_{kij}$; the probability of transition from state $i$ to $j$ concerning component $k$. The maximization is formulated in Eq. 2 subject to $\sum_{j=1}^{J} A_{kij} = 1$.

$$\max_{A_{kij}} \quad \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{m=2}^{M} \sum_{i=1}^{J} \sum_{j=1}^{J} \varepsilon_k(c_{m-1,i}^n, c_{m,j}^n) \log A_{kij} \tag{2}$$

**Algorithm 1.** MHMM

```
procedure LEARN(Y_{1:N}):
Initialise θ := {ρ, O, A, π}
    repeat
        E-step: calculate Q(θ, θ^old) := E_{C,Z|Y_{1:N},θ^old}[ log p(Y_{1:N}, C, Z|θ)]
        M-step: update A by Eq. 2
                update ρ, O, π
    until convergence
return θ
```

The EM algorithm concerning an MHMM is illustrated in Algorithm 1. For writing simplicity, we use MHMM when referring to this algorithm.

### 2.3 Orthogonality

In linear algebra, two vectors, $\mathbf{a}$ and $\mathbf{b}$, in a vector space are orthogonal when, geometrically, the angle between the vectors is 90 degrees. Equivalently, their inner product is zero, i.e. $\langle \mathbf{a}, \mathbf{b} \rangle = 0$. Similarly, the inner product of two orthogonal matrices is also zero. For matrices $A$ and $B$, $\langle A, B \rangle := trace(A^T B)$ [14] where *trace* refers to the sum of the elements of the diagonal of a matrix. The inner product of square matrices, $trace(A^T B)$, is calculated as the following [14]:

$$trace(A^T B) = \sum_{i=1}^{J} \sum_{j=1}^{J} A_{ij} B_{ij} \quad where \quad A, B \in R^{J \times J} \tag{3}$$

## 3 Orthogonal Mixture of Hidden Markov Models

In this section we describe our solution, Orthogonal Mixture of Hidden Markov Models (oMHMM), to avoid a poor local optimum in the MHMM. In a mixture of HMMs, each HMM, and consequently each transition matrix, corresponds to a different cluster. Therefore, we consider a transition matrix as a cluster representation. The underlying idea of the oMHMM method is to direct the search for a transition matrix solution away from candidate solutions that are very similar, i.e., not fully exploiting the power of the mixture model. This is achieved by increasing the distance between transition matrices—equivalently, the dissimilarity of the clusters—at each iteration of EM. A geometric intuition of the idea is illustrated in Fig. 1, where we use orthogonality as a distance measure between cluster representations (transition matrices). Intuitively, as the angle between the representations of the two clusters increases, the dissimilarity of those clusters increases.

In order to enforce the concept of distant transition matrices, we propose a penalty, *orthogonality penalty*, for the objective function described in Eq. 2 in the $M$-step of the EM algorithm. Concretely, the penalty is the sum of all the pairwise inner products of the transition matrices. By adding the orthogonality penalty, having a form as in Eq. 3, to the original maximization problem in Eq. 2, we achieve the penalized objective function presented in Eq. 4 subject to $\sum_{j=1}^{J} A_{kij} = 1$. By introducing a subtraction of the penalty, maximizing the objective function implies minimizing the inner product; i.e. maximizing the orthogonality. Note that $\lambda$ is the hyperparameter for the penalty.

$$\max_{A_{kij}} \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{m=2}^{M} \sum_{i=1}^{J} \sum_{j=1}^{J} \left[ \varepsilon_k(c^n_{m-1,i}, c^n_{m,j}) \log A_{kij} - \lambda \sum_{k' \neq k}^{K} trace(A^T_{kij} A_{k'ij}) \right] \tag{4}$$

Plugging the right hand side of Eq. 3 into Eq. 4, we get the following:

$$\max_{A_{kij}} \sum_{k=1}^{K} \sum_{n=1}^{N} \sum_{m=2}^{M} \sum_{i=1}^{J} \sum_{j=1}^{J} \varepsilon_k(c^n_{m-1,i}, c^n_{m,j}) \log A_{kij} - \lambda \sum_{k' \neq k}^{K} \sum_{i=1}^{J} \sum_{j=1}^{J} A_{kij} A_{k'ij} \tag{5}$$

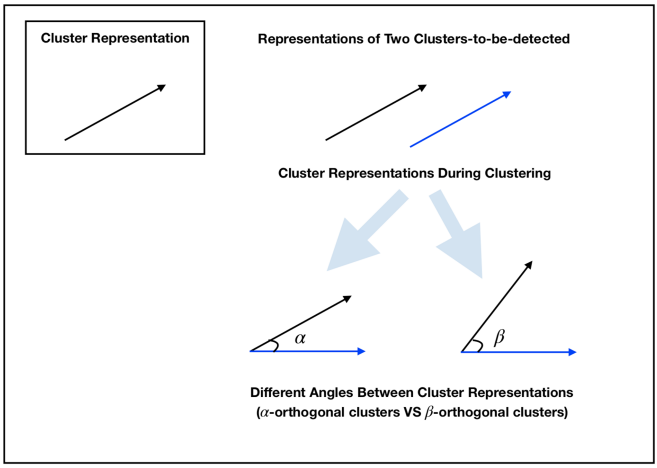


**Fig. 1.** A geometric view of the orthogonality of representations in clustering: each vector corresponds to a cluster representation, which can be expressed as any linear algebraic subspace. As the angle between representations of the two clusters increases, so does the dissimilarity between those clusters.

We solve the maximization problem stated in Eq. 5 by using the "cvxpy" Python library [2,11], as numerical optimization methods are needed due to the non-closed form solution to this optimization problem. Namely, setting the derivative of Eq. 5 w.r.t. $A_{kij}$, $\sum_{n=1}^{N}\sum_{m=2}^{M}\frac{\varepsilon(c^{k,n}_{m-1,i},c^{k,n}_{m,j})}{A_{kij}}+\delta-\sum_{k'\neq k}^{K}A_{k'ij}$, to zero cannot be solved by isolating the variable $A_{kij}$, where $\delta$ is the Lagrange multiplier of the constraint $\sum_{j=1}^{J}A_{kij}=1$.

A key feature of the oMHMM, lacking in the MHMM due to the treatment of transition matrices as independent, where the occurrence of one does not affect the probability of the occurrence of another, is that the estimation of a transition matrix, $A_k$, will be affected by all other transition matrices, $A_{k'} \quad \forall k' \neq k$; hence realizing a *global context*—parameters of *all* clusters are pulled into context.

Algorithm 2 summarizes oMHMM, which is a penalized EM. Note that all of the calculations are identical to the MHMM (Algorithm 1), except for line 5, where the update of transition matrices, $A$, is affected by Eq. 5.

## 4 Experiments

Our experimental objective is twofold: firstly, we aim to investigate the relative performance of the MHMM and oMHMM methods (Algorithms 1, 2) on various real-world datasets; we then use simulated datasets, inspired by the real-world datasets, to study the behavior of oMHMM in a more controlled manner, where the ground truth transition matrices are known.

**Algorithm 2.** oMHMM

**procedure** LEARN($Y_{1:N}$):
Initialise $\theta := \{\rho, O, A, \pi\}$
  **repeat**
    E-step: calculate $Q(\theta, \theta^{old}) := E_{C,Z|Y_{1:N},\theta^{old}}\left[\log p(Y_{1:N}, C, Z|\theta)\right]$
    M-step: update $A$ by Eq. 5
      update $\rho$, $O$, $\pi$
  **until** convergence
**return** $\theta$

For the implementation of oMHMM and the complete test results, we refer to https://github.com/negar7918/oMHMM.

### 4.1 Performance Metrics

When measuring method performance, we use v-measure [26] concerning clustering results (in all of the experiments performed, we have access to the cluster labels). The v-measure, which gives a score between 0% (imperfect) and 100% (perfect), captures the homogeneity and completeness properties, that is, it measures how well a cluster contains only its own members and how many of those members. Note that the rand index [25] provides very similar results as the v-measure, and is for that reason not included here.

For the real-world datasets, to extend the investigation of clustering performance, we use accuracy in addition to v-measure; accuracy is commonly used [15,30] despite the focus on unsupervised learning. Using accuracy, we can show the proportion of correctly estimated clusters. Moreover, we extend the investigation by reporting the number of EM iterations, measuring the computational cost (the elapsed time for each iteration of oMHMM and MHMM are nearly equal). As the goal of experimenting on the simulated datasets is to examine the clustering performance hypothesis, i.e., an increase of the orthogonality between the true transition matrices leads to increased clustering performance, it is sufficient to use v-measure.

The experiments are conducted using different random initializations, where each initialization is used by both of the methods, MHMM and oMHMM. In the tables below, the result of the better performing model is highlighted with a bold font.

### 4.2 Experiments with Biological Data

We perform experiments on a previously published biological dataset[2] stemming from single-cell whole-genome sequencing from 18 primary colon cancer tumor cells and 18 metastatic cells from matched liver samples for one patient

[2] Available from the NCBI Sequence Read Archive (SRA) under accession number SRP074289; for pre-processing of the data, see [18].

referred to as CRC2 in [18]. In this work, we cluster CRC2 patient data using the genomic sequence of chromosome 4; the sequence comprises 808 genomic regions (sequence length of 808) where each region bears the characteristic of that region by a count number. In other words, the nature of the data is a count number. As mentioned in [18], it is reasonable that the primary tumor cells from the colon and the metastatic cells from the liver should cluster separately; therefore, we consider to cluster the sequence data into primary and metastatic clusters. Note that, for the purpose of evaluation, we know which cell sequence belongs to which cluster according to [18]. In order to perform clustering, we use a mixture of HMMs, where each cell sequence with a length of 808 represents an observation sequence and each HMM comprises hidden variables with three states, where the states correspond to the *copy numbers* for chromosome 4 used in [18]. In Fig. 2, illustrating metastatic and primary cancer cells, we can observe how, on average, the depth of the count data concerning metastatic cells reach lower values than the case of primary cells. Moreover, we can see that the variation of the counts across the sequence position is higher in metastatic than primary cells. The variations occurring throughout the sequence data are commonly used to perform clustering in cancer research [18]. These variations, per cluster, can be modeled as state transitions in an HMM in the mixture model. In Fig. 3, two cells are shown, each belonging to a different cluster: M-67 from the metastatic cluster and P-8 from the primary cluster. Note that, it is harder to see their difference compared to the difference between the average of metastatic and primary cells illustrated in Fig. 2. However, we can still see that they follow the patterns in Fig. 2; i.e., the lower counts belong to the metastatic cell (M-67) and the variation of the counts is higher in M-67 compared to P-8.

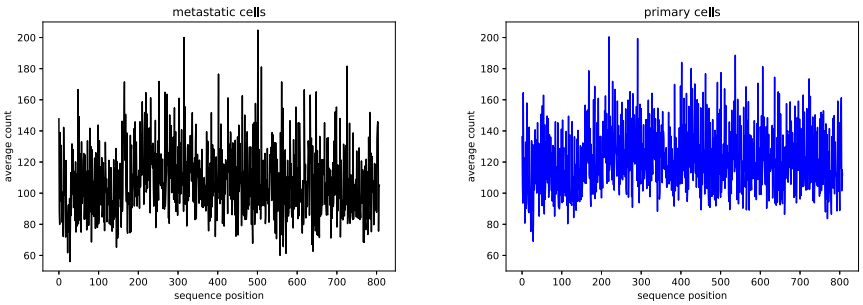


**Fig. 2.** The average counts of metastatic and primary cancer cells, per sequence position.

After performing the clustering of metastatic- and primary cells, using the MHMM and oMHMM methods, we calculate the resulting v-measure and accuracy measures. Moreover, we give an account of the number of EM iterations for each method. We run the methods on the random initializations A and B concerning transition matrices; initialization A has a Dirichlet distribution prior

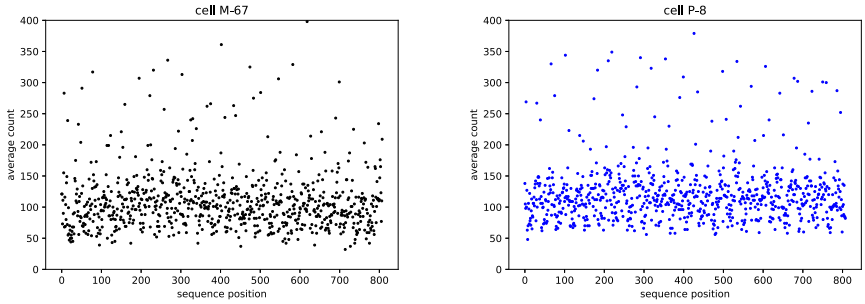


**Fig. 3.** The count data concerning two cells: M-67 metastatic and P-8 primary.

with parameters set to 0.1 for each row of the transition matrix, and B follows a discrete uniform distribution (the choices of the priors are inspired by [3,30]). We set the orthogonality penalty's hyperparameter to 1, allowing for a full effect of the orthogonality penalty. Table 1 shows that the oMHMM outperforms the MHMM for both initializations w.r.t. v-measure, accuracy, and the number of iterations. The highest v-measure and accuracy are 84% and 97%, respectively, and the maximum improvement is 55 percentage points w.r.t. v-measure and 44 percentage points w.r.t. accuracy, for initialization A. Finally, looking at the table concerning the number of iterations, we can observe that the oMHMM outperforms the MHMM with one iteration fewer required by the oMHMM than that for the MHMM. The fewer number of iterations implies a faster convergence.

**Table 1.** Accuracy, v-measure, and number of iterations are compared between the MHMM and oMHMM.

% V-measure

| initialization | MHMM | oMHMM |
|---|---|---|
| A | 4% | **59**% |
| B | 69% | **84**% |

% Accuracy

| initialization | MHMM | oMHMM |
|---|---|---|
| A | 47% | **91**% |
| B | 94% | **97**% |

# Iterations

| initialization | MHMM | oMHMM |
|---|---|---|
| A | 5 | **4** |
| B | 3 | **2** |

### 4.3 Experiments with Handwritten Digit Data

We cluster handwritten digits from the so-called "pen-based recognition of handwritten digits" dataset in the UCI machine learning repository [4]. We repeatedly cluster datasets obtained by restricting the entire dataset to two digits at a time

and considering two clusters similar to [30]; despite of oMHMM being capable of multi-class clustering. In each such experiment, we use two and four hidden states in the MHMM and oMHMM. The orthogonality penalty hyperparameter is, as previously, set to 1 and the random initializations A and B from the previous section are used. First, using initialization A, we compare the MHMM and oMHMM performance, see Table 2. In the v-measure and accuracy tables, the MHMM and oMHMM are compared per number of hidden states e.g., the oMHMM outperforms the MHMM w.r.t. accuracy (83% vs. 43%) on the "digit 6 vs 7" dataset when using four hidden states. We make the following two observations concerning the results presented in Table 2: (i) the oMHMM outperforms the MHMM and achieves the v-measures 36%, 29%, and 7%, corresponding to accuracies 83%, 76%, and 59%; (ii) the total number of iterations is reduced approximately to one third for all of the datasets. As shown in Table 3 for initialization B, the oMHMM outperforms the MHMM with v-measures 91%, 38%, 28%, 18%, and 10% corresponding to accuracies 98%, 78%, 73%, 68%, and 60%. For all of the datasets, the total number of iterations for the oMHMM is only 20% of that for the MHMM.

For each of the four datasets ("digit 6 vs 7", "digit 2 vs 9", "digit 4 vs 2", and "digit 5 vs 8"), the oMHMM results in the best v-measure and accuracy: (i) the highest v-measure are 91%, 29%, 18%, and 10%, respectively; (ii) the accuracies are 98%, 76%, 68%, and 60%, respectively. Note that in each of the experiments, the oMHMM results in a v-measure and accuracy that are equal to or greater than those of the MHMM. The greatest v-measure improvement obtained by the oMHMM is 38 percentage points increment for the dataset "digit 6 vs 7" and two states, Table 3. As shown in Table 2, the oMHMM results in the greatest accuracy increase, 40 percentage points increment, for "digit 6 vs 7" and four states. Finally, the oMHMM outperforms the MHMM with respect to the number of iterations, reducing the computational cost to approximately 20–50% of that achieved by the MHMM.

### 4.4 Experiments with Hand Movement Data

We perform experiments on the Libras movement dataset from the UCI machine learning repository [4], in which there are 15 classes of hand movements. Experiments are performed similarly to the previous section. We repeatedly cluster datasets obtained by restricting the entire dataset to two hand movements at a time.

First, using initialization A, we compare the MHMM and oMHMM performance, see Table 4. We can see that the oMHMM outperforms the MHMM and achieves v-measures 34%, 3%, 34%, 13%, 1%, and 17% corresponding to accuracies 75%, 60%, 75%, 71%, 54%, and 62%. Note that the accuracy achieved by the oMHMM is greater than the one from the MHMM for the "2 vs 1" dataset when using two hidden states; however, the v-measure is the same for both methods. The total number of iterations is reduced for the "13 vs 15" dataset and is almost unchanged for the other datasets.

**Table 2.** Using initialization A, accuracy, v-measure, and number of iterations are compared between the MHMM and oMHMM. S shows the number of hidden states.

% V-measure

| Dataset | MHMM | | oMHMM | |
|---|---|---|---|---|
| | S=2 | S=4 | S=2 | S=4 |
| digit 6 vs 7 | 5% | 2% | 5% | **36**% |
| digit 2 vs 9 | 21% | 0% | **29**% | 0% |
| digit 4 vs 2 | 0% | 0% | **7**% | 0% |
| digit 5 vs 8 | 1% | 0% | 1% | 0% |

% Accuracy

| Dataset | MHMM | | oMHMM | |
|---|---|---|---|---|
| | S=2 | S=4 | S=2 | S=4 |
| digit 6 vs 7 | 37% | 43% | 37% | **83**% |
| digit 2 vs 9 | 73% | 47% | **76**% | 47% |
| digit 4 vs 2 | 50% | 50% | **59**% | 50% |
| digit 5 vs 8 | 44% | 50% | 44% | 50% |

# Total Iterations

| Dataset | MHMM | oMHMM |
|---|---|---|
| digit 6 vs 7 | 16 | **5** |
| digit 2 vs 9 | 17 | **5** |
| digit 4 vs 2 | 17 | **5** |
| digit 5 vs 8 | 16 | **6** |

**Table 3.** Using initialization B, accuracy, v-measure, and number of iterations are compared between the MHMM and oMHMM. S shows the number of hidden states.

% V-measure

| Dataset | MHMM | | oMHMM | |
|---|---|---|---|---|
| | S=2 | S=4 | S=2 | S=4 |
| digit 6 vs 7 | 0% | 60% | **38**% | **91**% |
| digit 2 vs 9 | 0% | 0% | **28**% | 0% |
| digit 4 vs 2 | 0% | 0% | **18**% | 0% |
| digit 5 vs 8 | 0% | 0% | **10**% | 0% |

% Accuracy

| Dataset | MHMM | | oMHMM | |
|---|---|---|---|---|
| | S=2 | S=4 | S=2 | S=4 |
| digit 6 vs 7 | 51% | 91% | **78**% | **98**% |
| digit 2 vs 9 | 48% | 48% | **73**% | 48% |
| digit 4 vs 2 | 50% | 50% | **68**% | 50% |
| digit 5 vs 8 | 50% | 50% | **60**% | 50% |

# Total Iterations

| Dataset | MHMM | oMHMM |
|---|---|---|
| digit 6 vs 7 | 28 | **15** |
| digit 2 vs 9 | 31 | **6** |
| digit 4 vs 2 | 30 | **11** |
| digit 5 vs 8 | 35 | **7** |

As shown in Table 5 for initialization B, the oMHMM outperforms the MHMM with v-measures of 13%, 2%, 13% and 1%; however, w.r.t. accuracy, the oMHMM outperforms the MHMM on more accounts with values 73%, 71%, 58%, 71%, 54%, and 52%. The total number of iterations is reduced for the datasets "15 vs 3" and "3 vs 1", while unchanged for the other two datasets.

For each of the four datasets ("13 vs 15", "15 vs 3", "3 vs 1", and "2 vs 1"), the oMHMM results in the best v-measure and accuracy: (i) the highest

**Table 4.** Using initialization A, accuracy, v-measure, and number of iterations are compared between the MHMM and oMHMM. S shows the number of hidden states.

% V-measure

| Dataset | MHMM | | oMHMM | |
|---|---|---|---|---|
| | S=2 | S=4 | S=2 | S=4 |
| 13 vs 15 | 1% | 1% | **3%** | **34%** |
| 15 vs 3 | 0% | 0% | 0% | **34%** |
| 3 vs 1 | 4% | 0% | **13%** | **1%** |
| 2 vs 1 | 5% | 0% | 5% | **17%** |

% Accuracy

| Dataset | MHMM | | oMHMM | |
|---|---|---|---|---|
| | S=2 | S=4 | S=2 | S=4 |
| 13 vs 15 | 54% | 54% | **60%** | **75%** |
| 15 vs 3 | 52% | 48% | 52% | **75%** |
| 3 vs 1 | 62% | 50% | **71%** | **54%** |
| 2 vs 1 | 60% | 50% | **62%** | **62%** |

# Total Iterations

| Dataset | MHMM | oMHMM |
|---|---|---|
| 13 vs 15 | 12 | **7** |
| 15 vs 3 | **8** | 9 |
| 3 vs 1 | **8** | 9 |
| 2 vs 1 | **8** | 9 |

**Table 5.** Using initialization B, accuracy, v-measure, and number of iterations are compared between the MHMM and oMHMM. S shows the number of hidden states.

% V-measure

| Dataset | MHMM | | oMHMM | |
|---|---|---|---|---|
| | S=2 | S=4 | S=2 | S=4 |
| 13 vs 15 | 0% | 20% | **13%** | 20% |
| 15 vs 3 | 2% | 0% | 2% | **2%** |
| 3 vs 1 | 0% | 6% | **1%** | **13%** |
| 2 vs 1 | 0% | 0% | 0% | 0% |

% Accuracy

| Dataset | MHMM | | oMHMM | |
|---|---|---|---|---|
| | S=2 | S=4 | S=2 | S=4 |
| 13 vs 15 | 50% | 64% | **71%** | **73%** |
| 15 vs 3 | 58% | 54% | 58% | **58%** |
| 3 vs 1 | 52% | 65% | **54%** | **71%** |
| 2 vs 1 | 50% | 50% | 50% | **52%** |

# Total Iterations

| Dataset | MHMM | oMHMM |
|---|---|---|
| 13 vs 15 | 14 | 14 |
| 15 vs 3 | 24 | **8** |
| 3 vs 1 | 18 | **14** |
| 2 vs 1 | 4 | 4 |

v-measures are 34%, 34%, 13%, and 17%, respectively; (ii) the accuracies are 75%, 75%, 71%, and 62%, respectively. Note that in each of the experiments, the oMHMM results in v-measure and accuracy of equal to or greater than those of the MHMM. The greatest improvement obtained by the oMHMM w.r.t. v-measure and accuracy concerns the "15 vs 3" dataset using four hidden states in Table 4, with an increase of 34 and 27 percentage points, respectively.

### 4.5 Experiments with Simulated Data

The goal of this section is to test the following hypothesis: greater orthogonality between the true transition matrices results in greater improvement achieved by the oMHMM compared to MHMM. This may only be investigated when having access to the ground truth (the true transition matrices), which is why we consider synthetic datasets here. We consider data generated from a given MHMM, comprising ground truth model parameters (we refer to this model by MHMM-gen), and evaluate the clustering obtained by each of the MHMM and the oMHMM from this data. The contribution of the orthogonality penalty is investigated by comparing the v-measures obtained by the oMHMM and the MHMM, based on the ground truth cluster labels from MHMM-gen.

Inspired by the biological data comprising two clusters in Sect. 4.2, we construct the simulated datasets, assuming the two transition patterns where each represents an inclination towards a specific state (we assume 3 vs. 4). Moreover, we consider 50 observation sequences with a length of 800. These sequences form the first dataset, *Scenario 1*. To study the behavior of the oMHMM when having lower orthogonality among the transition matrices in the ground truth model, we create a new dataset, *Scenario 2*, where we design the transition matrices so that they result in lower orthogonality than in Scenario 1. We achieve this by adding a third HMM (cluster) following a transition pattern similar to one of the two transition patterns in Scenario 1; this similarity results in lower orthogonality.

Having the ground truth, we evaluate the clustering results produced by the MHMM and oMHMM. V-measure is used as the main clustering performance metric. We perform the tests considering the transition matrix initializations used in the real-world datasets, A and B, to evaluate the orthogonality hypothesis in the already performed setting. Moreover, we add a third initialization, C, to extend the evaluation. C holds a Dirichlet distribution prior with parameters set to 0.5 for each row of the transition matrix.

Regarding the hyperparameters of the orthogonality penalty, $\lambda$ in Eq. 5, we define the set of possible values to be $\{0, 0.1, 0.5, 1\}$. For each scenario and each initialization, we tune the value of $\lambda$ using a separate dataset (with the same size as the test datasets used subsequently in the experiments) which we never use again in the following experiments. For each of those separate datasets, we choose the $\lambda$ which gives the highest v-measure when performing the oMHMM. In case of ties among the best-performing values of $\lambda$, the highest value is selected.

**Scenario 1.** Using MHMM-gen, we generate 50 observation sequences of length 800, divided into two clusters. The parameters of the MHMM-gen model are set as follows. The number of hidden states is set to 4, the emission distribution is a Poisson distribution with one state-dependent rate per hidden state, randomly chosen between 80 and 100. We use mixture probabilities of 0.5 for the two components with transition matrices, $A_1$ and $A_2$. Conceptually, one cluster is inclined to stay at state 3 and the other at state 4 (these are highlighted as column 3 and 4 in the matrices below). In order to compare Scenario 1 to Scenario 2 for testing our hypothesis, we use orthogonality which is defined as one minus

**Table 6.** V-measure is compared between MHMMs and oMHMMs for Scenario 1 and 2.

% V-measure Scenario 1

| initialization | MHMM | oMHMM |
|---|---|---|
| A | 29% | 29% |
| B | 40% | **87**% |
| C | 0% | **11**% |

% V-measure Scenario 2

| initialization | MHMM | oMHMM |
|---|---|---|
| A | 72% | 72% |
| B | 62% | **72**% |
| C | 66% | **72**% |

the normalized inner product, i.e., $1 - \frac{<A_1,A_2>}{\sqrt{<A_1,A_1>}\sqrt{<A_2,A_2>}}$. The orthogonality of the matrices, $A_1$ and $A_2$, is 89%.

$$A_1 = \begin{pmatrix} 0 & 0 & .07 & \mathbf{.93} \\ 0 & .003 & .007 & \mathbf{.99} \\ 0 & 0 & .06 & \mathbf{.94} \\ 0 & 0 & .02 & \mathbf{.98} \end{pmatrix} \quad A_2 = \begin{pmatrix} 0 & .002 & \mathbf{.99} & .008 \\ 0 & 0 & \mathbf{.95} & .05 \\ 0 & 0 & \mathbf{.92} & .08 \\ 0 & 0 & \mathbf{.87} & .13 \end{pmatrix}$$

The hyperparameter values of the orthogonality penalty, $\lambda$, for initializations A, B, and C are 0, 1, and 1, respectively, after performing hyperparameter tuning as aforementioned. We give account for the v-measure performance of the MHMM and oMHMM concerning different initializations in the left-hand side of Table 6. We can observe that the oMHMM outperforms the MHMM with a 47 and 11 percentage points increase in v-measure for initialization B and C, respectively. Finally, the oMHMM results in the highest v-measure.

**Scenario 2.** Similar to Scenario 1, we generate observation sequences (a total of 100), forming three clusters. Each HMM is generated similarly to Scenario 1, except that the sequence length is 200 and the four Poisson rates are set to 1, 3, 9, and 27, respectively, for states 1, 2, 3, and 4. The following transition matrices are considered for the three HMMs: $A_1$, $A_2$, and $A_3$. $A_1$ and $A_2$ follow Scenario 1 and $A_3$ is similar to $A_1$, however with less inclination towards state 4. The orthogonality of 65% is achieved, which is 24 percentage points less than that for Scenario 1 (the orthogonality score is calculated using the same formula as in Scenario 1).

$$A_1 = \begin{pmatrix} 0 & 0 & .07 & \mathbf{.93} \\ 0 & .003 & .007 & \mathbf{.99} \\ 0 & 0 & .06 & \mathbf{.94} \\ 0 & 0 & .02 & \mathbf{.98} \end{pmatrix} \quad A_2 = \begin{pmatrix} 0 & .002 & \mathbf{.99} & .008 \\ 0 & 0 & \mathbf{.95} & .05 \\ 0 & 0 & \mathbf{.92} & .08 \\ 0 & 0 & \mathbf{.87} & .13 \end{pmatrix} \quad A_3 = \begin{pmatrix} 0 & .3 & 0 & .7 \\ 0 & .4 & 0 & .6 \\ 0 & .3 & 0 & .7 \\ 0 & .3 & .02 & .68 \end{pmatrix}$$

The hyperparameter $\lambda$ is again tuned using the procedure outlined for Scenario 1. For initialization A, $\lambda = 1$ is chosen as it results in the highest v-measure. For initializations B and C, due to the identical v-measure results, $\lambda = 1$ is chosen (following our assumption explained in Sect. 4.5).

Looking at the right-hand side in Table 6, we can observe that the oMHMM outperforms the MHMM with a 10 and 6 percentage points increase in v-measure regarding initialization B and C, respectively. Note that the oMHMM results in the highest v-measure; however, the oMHMM has equal performance to MHMM, using initialization A. Comparing these results to the ones from Scenario 1, we can observe that the contribution of the oMHMM decreases as the orthogonality among transition matrices decreases. This confirms our hypothesis that the more orthogonal the ground truth transition matrices are, the greater improvement can be expected from the oMHMM.

### 4.6 Discussion of Results

The experiments showed that oMHMM can significantly improve MHMM. Concretely, oMHMM was observed to achieve up to a 55 percentage points increase w.r.t. v-measure, a 44 percentage points increase w.r.t. accuracy, and was observed to reduce the number of iterations down to a fifth. The experiments conducted on simulated data confirmed our hypothesis that the more orthogonal the ground truth transition matrices are, the greater improvement may be obtained by using the oMHMM instead of the standard MHMM. When the orthogonality is less pronounced we expect oMHMM to perform equal to MHMM given that the penalty hyperparameter is properly tuned.

## 5 Concluding Remarks

The use of EM for clustering of sequential data based on an MHMM may lead to poor local optima. This type of problem is often handled by augmenting the objective function, in the $M$-step of EM, with a penalty term. Several different penalties have been proposed for the EM algorithm when handling a single HMM. To tackle the problem for an MHMM, we propose a new penalty, the orthogonality penalty, which takes multiple HMMs into account. We call the so obtained EM algorithm oMHMM. The underlying idea is that the clustering can be expected to be improved when increasing the dissimilarity of clusters based on the orthogonality of the corresponding transition matrices of the constituent HMMs. We have presented results from experiments in which the novel algorithm is compared to the standard EM for an MHMM. The results show that the oMHMM performs on par or better than the standard EM for an MHMM with respect to v-measure, accuracy, and the number of iterations. These promising results show that the proposed penalty has a positive effect on sequence clustering using an MHMM.

One direction for future research is to combine the sparse mixture of HMMs [24] with the oMHMM. Another direction concerns investigating the theoretical and statistical properties of the oMHMM, e.g., consistency, efficiency, and convergence rate. MHMMs, in contrast to HMMs, have not received sufficient attention, e.g., various penalties can be investigated concerning EM for an MHMM. Finally, observing the high performance of oMHMM on the biological data in this

work, motivates future work on critical biomedical applications. To name some, one can extend the studies on cancer cell clustering [28] and early prediction of a therapy outcome [27] by applying the orthogonality constraint.

**Acknowledgments.** We thank Johan Fylling, Mohammadreza Mohaghegh Neyshabouri, and Diogo Pernes for their great help during the preparation of this paper.

## References

1. Aghabozorgi, S., Seyed Shirkhorshidi, A., Ying Wah, T.: Time-series clustering - a decade review. Inf. Syst. **53**, 16–38 (2015)
2. Agrawal, A., Verschueren, R., Diamond, S., Boyd, S.: A rewriting system for convex optimization problems. J. Control Decision **5**(1), 42–60 (2018)
3. Altosaar, J., Ranganath, R., Blei, D.: Proximity variational inference. AISTATS (2017)
4. Bache, K., Lichman, M.: Uci machine learning repository. UCI machine learning repository (2013)
5. Baum, L., Petrie, T.: Statistical inference for probabilistic functions of finite state markov chains. Ann. Math. Stat. **37**(6), 1554–1563 (1966)
6. Bishop, C.: Pattern recognition and machine learning. Springer, Information science and statistics, New York (2006)
7. Bishop, C.: Model-based machine learning. Philosophical transactions. Series A, Mathematical, physical, and engineering sciences 371 (2012)
8. Blei, D., Kucukelbir, A., Mcauliffe, J.: Variational inference: a review for statisticians. J. Am. Statist. Assoc. **112**(518), 859–877 (2017)
9. Chamroukhi, F., Nguyen, H.: Model based clustering and classification of functional data. Wiley Interdiscip. Rev. Data Mining Knowl. Disc. **9**(4), e1298 (2019)
10. Dempster, A.P., Laird, N.M., Rubin, D.B.: Maximum likelihood from incomplete data via the EM algorithm. J. Roy. Stat. Soc.: Ser. B (Methodol.) **39**(1), 1–22 (1977)
11. Diamond, S., Boyd, S.: CVXPY: a Python-embedded modeling language for convex optimization. J. Mach. Learn. Res. **17**(83), 1–5 (2016)
12. Dias, J., Vermunt, J., Ramos, S.: Mixture hidden markov models in finance research. In: Advances in Data Analysis, Data Handling and Business Intelligence, pp. 451–459 (2009)
13. Esmaili, N., Piccardi, M., Kruger, B., Girosi, F.: Correction: Analysis of healthcare service utilization after transport-related injuries by a mixture of hidden markov models. PLoS One **14**(4), e0206274 (2019)
14. Horn, R.A., Johnson, C.R.: Matrix Analysis. Cambridge University Press, Cambridge (2013)
15. Jebara, T., Song, Y., Thadani, K.: Spectral clustering and embedding with hidden markov models. In: Machine Learning: ECML 2007: 18th European Conference on Machine Learning 4701, pp. 164–175 (2007)
16. Jonathan, A., Sclaroff, S., Kollios, G., Pavlovic, V.: Discovering clusters in motion time-series data. In: CVPR (2003)
17. Kulesza, A., Taskar, B.: Determinantal point processes for machine learning. Found. Trends Mach. Learn. **5**(2–3), 123–286 (2012)
18. Leung, M., et al.: Single-cell DNA sequencing reveals a late-dissemination model in metastatic colorectal cancer. Genome Res. **27**(8), 1287–1299 (2017)

19. Ma, Q., Zheng, J., Li, S., Cottrell, G.: Learning representations for time series clustering. Adv. Neural Inf. Process. Syst. **32**, 3781–3791 (2019)
20. Maoying Qiao, R., Bian, W., Xu, D., Tao, D.: Diversified hidden markov models for sequential labeling. IEEE Trans. Knowl. Data Eng. **27**(11), 2947–2960 (2015)
21. McGibbon, R., Ramsundar, B., Sultan, M., Kiss, G., Pande, V.: Understanding protein dynamics with l1-regularized reversible hidden markov models. In: Proceedings of the 31st International Conference on Machine Learning, vol. 32, no. 2, pp. 1197–1205 (2014)
22. Montanez, G., Amizadeh, S., Laptev, N.: Inertial hidden markov models: modeling change in multivariate time series. In: AAAI Conference on Artificial Intelligence (2015)
23. Oates, T., Firoiu, L., Cohen, P.: Clustering time series with hidden markov models and dynamic time warping. In: IJCAI-99 Workshop on Neural, Symbolic and Reinforcement Learning Methods for Sequence Learning, pp. 17–21 (1999)
24. Pernes, D., Cardoso, J.S.: Spamhmm: sparse mixture of hidden markov models for graph connected entities. In: 2019 International Joint Conference on Neural Networks (IJCNN), pp. 1–10 (2019)
25. Rand, W.: Objective criteria for the evaluation of clustering methods. J. Am. Statist. Assoc. **66**(336), 846–850 (1971)
26. Rosenberg, A., Hirschberg, J.: V-measure: a conditional entropy-based external cluster evaluation measure. In: EMNLP-CoNLL (2007)
27. Safinianaini, N., Boström, H., Kaldo, V.: Gated hidden markov models for early prediction of outcome of internet-based cognitive behavioral therapy. In: Riaño, D., Wilk, S., ten Teije, A. (eds.) AIME 2019. LNCS (LNAI), vol. 11526, pp. 160–169. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-21642-9_22
28. Safinianaini, N., De Souza, C., Lagergren, J.: Copymix: mixture model based single-cell clustering and copy number profiling using variational inference. bioRxiv (2020). https://doi.org/10.1101/2020.01.29.926022
29. Smyth, P.: Clustering sequences with hidden markov models. In: Advances in Neural Information Processing Systems (1997)
30. Subakan, C., Traa, J., Smaragdis, P.: Spectral learning of mixture of hidden markov models. Adv. Neural Inf. Process. Syst. **27**, 2249–2257 (2014)
31. Tao, L., Elhamifar, E., Khudanpur, S., Hager, G., Vidal, R.: Sparse hidden markov models for surgical gesture classification and skill evaluation. In: Proceedings of International Conference on Natural Language Processing and Knowledge Engineering, pp. 167–177 (2012)
32. Wang, Q., Schuurmans, D.: Improved estimation for unsupervised part-of-speech tagging. In: Proceedings of International Conference on Natural Language Processing and Knowledge Engineering, pp. 219–224 (2005)
33. Xing, Z., Pei, J., Keogh, E.: A brief survey on sequence classification. ACM SIGKDD Explor. Newslett. **12**(1), 40–48 (2010)
34. Yuting, Q., Paisley, J., Carin, L.: Music analysis using hidden markov mixture models. IEEE Trans. Signal Process. **55**(11), 5209–5224 (2007)