Noname manuscript No.
(will be inserted by the editor)

DBHC: Discrete Bayesian HMM Clustering

Received: date / Accepted: date

Abstract Sequence data mining has become an increasingly popular research topic as the availability of data has grown rapidly over the past decades. Sequence clustering is a type of method within this field that is in high demand in the industry, but the sequence clustering problem is non-trivial and, as opposed to static cluster analysis, interpreting clusters of sequences is often difficult. With the Discrete Bayesian HMM Clustering (DBHC) algorithm, we propose an approach to clustering discrete sequences using hidden Markov models (HMMs) by extending a proven method for continuous sequences. Our algorithm is completely self-contained as it incorporates both the search for the number of clusters and the search for the number of hidden states in each cluster model in the parameter inference. By means of an illustrative example, we show how the hidden states in a mixture of HMMs can aid the interpretation task of a sequence cluster analysis. We conclude that the algorithm works well as it provides well-interpretable clusters for our application.

Keywords Sequence Data Mining · Sequence Clustering · Mixture Hidden Markov Models · Graphical Models · Probability Smoothing

1 Introduction

This paper provides an approach to clustering discrete sequences using hidden Markov models (HMMs). We propose the Discrete Bayesian HMM Clustering (DBHC) algorithm, an extension of the Bayesian HMM Clustering algorithm to discrete sequence data (Li and Biswas, 2000). The algorithm incorporates the searches for the number of clusters and for the number of hidden states in each cluster model, two classical problems that one is faced with when working with mixture models and HMMs. The algorithm features a procedure that initialises cluster models by evaluating them on a small set (seed) of observations, which is called the 'seed selection procedure'. By adding a probability smoothing step in this seed selection procedure, we extend the framework of Li and Biswas (2000) to

fit discrete sequence data. To the best of our knowledge, this framework has never been extended to this domain. Additionally, we propose an implementation of the DBHC algorithm in the R statistical software (R Core Team, 2017).

Clustering sequence data is inherently different than clustering static data. One of the main reasons for this is the intransitivity of the problem. For example, if we regard two sequences `AAAA` and `BBB` to be similar to `AAAABBB` it does not mean that `AAAA` is also similar to `BBB` (Dong and Pei, 2007). Intransitivity makes the sequence clustering problem non-trivial. However, by using an adjusted distance function that incorporates the sequential structure it becomes possible to use traditional clustering methodology for sequence data. A special type of distance function is the probabilistic distance function: using a probability distribution to describe differences between clusters, which is the main idea behind mixture models (in general). Using a mixture of HMMs, the model is also able to account for differences in sequence-time variation across clusters.

Because of the dimensionality of the problem and the dynamic nature of sequence data, it is often difficult to interpret clusters of sequences. This is a major drawback for analysis of this data type, because the main goal of cluster analysis is usually description, not prediction. For the purpose of solving this problem, visualising cluster patterns in sequence data might aid the interpretation task. Cadez et al. (2003) consider sequences where the observations are assumed to develop according to a first-order Markov process, i.e., observations are assumed to be drawn from a distribution that only depends on the previous observation in a sequence. They developed a tool that visualises discrete sequence observations for each cluster using colour coding. In this case the sequential ordering in the data is not a drawback of the problem setting anymore, it helps interpreting the clusters. Clustering sequences using multiple instances of an HMM may also aid the interpretation task. An HMM allows for modelling dynamic processes with an unobserved variable that traverses through different states, which are called 'hidden' states. In this model, not the observations, but the hidden states are assumed to follow a first-order Markov process. Visualising the estimated probabilities of such a model in heatmaps can also help understanding differences between clusters.

We employ a mixture of HMMs for clustering discrete sequence data with the DBHC algorithm. In a mixture of HMMs, both the number of clusters and the number of states for each cluster are usually unknown. We follow the approach of Li and Biswas (2000) and incorporate the searches for these unknown parameters in the model by casting them as Bayesian model selection problems. In Li and Biswas (2000), clusters are iteratively added to the total partition, while each new cluster initially contains a carefully selected set of observations, called a 'seed', before other observations are distributed over the clusters. Since the method of Li and Biswas (2000) is intended for continuous observations only, we extend the seed selection procedure to be applicable for discrete observations and propose the Discrete Bayesian HMM Clustering (DBHC) algorithm.

The rest of this paper is structured as follows. First, in Section 2 we describe the related work in this field. Second, in Section 3 we describe the model underlying the mixture used for sequence clustering in the DBHC algorithm: the discrete-output HMM. Then follows a detailed description of the methodology of the DBHC algorithm in Section 4. In Section 5, we discuss an example of sequence clustering with the DBHC algorithm. Lastly, we summarise our findings and provide concluding remarks in Section 6.

2 Related Work

In current literature, most applications of sequence clustering can be found in the fields of biology and Web mining. Dong and Pei (2007) provide a complete overview of the approaches to sequence clustering in the current literature. They argue that it is important to select a distance function that incorporates the sequential structure of the data, and next, to select an algorithm that is suitable for this distance function. They conclude that the choice for the distance function should be based on domain knowledge of the subject of interest. A popular algorithm that naturally fits a wide range of distance functions is the hierarchical clustering algorithm. For example, the authors of Burke et al. (1999) adapt the single-linkage hierarchical clustering algorithm to cluster DNA sequences by using a distance function that is suitable for DNA sequences. They conclude that their algorithm works well for their application in the field of DNA research.

It is also possible to cluster sequences using a model-based approach, which is making use of a probabilistic distance function. Cadez et al. (2003) present a mixture of first-order Markov models for clustering visitors of a Web page, where the sequence data is assumed to be generated by a Markov model. They also present a visualisation tool for the clusters, which aids interpretation tasks. They state that a mixture of first-order models is not a first-order model, because it can model much more flexible relations. However, their method is static and assumes constant transition probabilities over time.

The usage of graphical models for the purpose of sequence clustering was first employed by Rabiner et al. (1989). They propose an algorithm for speech recognition that makes use of clustering using an HMM with continuous output. The problem of clustering speech data is inherently dynamic when audio is logged over time. They are the first to use an HMM in a clustering problem, which they do by estimating an HMM for each cluster, known as a finite mixture of HMMs. An advantage of formulating the problem in this way is that the clustering can now be incorporated in the HMM parameter estimation procedure. The authors propose two methods for obtaining clusters in an HMM. The first is incorporating the clustering in the likelihood function, which can then easily be maximised. However, they conclude that this method appears to not work well in practice, as it only improves the fit for observations who were already represented well by an initial model in the procedure. The second method, which works much better, as concluded by the authors, is a cluster splitting procedure. In this procedure, clusters are iteratively split into smaller clusters, until a threshold is reached in the likelihood objective. In such a split they fit an extra HMM to previously poorly represented observations. A major disadvantage of their work is that the threshold can usually only be based on a simple guess, and another disadvantage is that the model performance is very sensitive for the chosen threshold.

Later on, the maximum likelihood approach to HMM clustering was improved by others, for example in Smyth (1997). The author extends the method in two ways, by incorporating hierarchical clustering for initialisation of the algorithm and by adding a cross-validation approach for determining the number of clusters. He also proposes to estimate the HMMs by using the *Baum-Welch* algorithm, which is a special case of the Expectation-Maximisation (EM) algorithm, and a common way to estimate ordinary HMMs (Rabiner, 1989). His approach does assume the number of clusters known, and the solution is to estimate the model for

different numbers of clusters and determine the optimal value with cross-validation afterwards. The author concluded in a simulation experiment that this search for the number of clusters works well. As a recommendation for further research he suggests to incorporate the search for the number of clusters in the estimation procedure using Bayesian statistics. We follow the advice of Smyth (1997) and cast the search for the number of clusters as a Bayesian model selection problem.

Bayesian statistics have also been introduced to estimating probabilities in an HMM, for example in Stolcke and Omohundro (1994). The authors use a Bayesian model-merging strategy for finding the number of states in an HMM. Nevertheless, they do not use the HMM for clustering and, therefore, assume a homogeneous population. They do consider an HMM with discrete output, for which they use a Dirichlet prior distribution, a multinomial extension of the Beta distribution. The authors do not extensively compare their prior distribution proposal to other alternatives, but they conclude that the prior type and parameters do not influence the search for the number of HMM states very much. They do admit, however, that including informative priors rather than flat (uninformative) priors will probably have an impact on the course of this search.

Li and Biswas (2000) are the first to employ a Bayesian approach for sequence clustering using HMMs. They consider HMMs with continuous output and do not require the number of states and number of clusters known. Furthermore, they do not assume that each HMM has the same model size, i.e., the HMMs do not need to have the same number of states. They propose an algorithm that alternately searches for the optimal number of clusters, cluster assignment of objects, optimal number of states, and the parameters of each HMM. Furthermore, they propose a careful seed selection procedure for initialising cluster models. They do not use a full Bayesian approach, however, as they infer parameters using the frequentist Baum-Welch algorithm and do not use a Bayesian mixture for the clustering part of the algorithm, but rather cast the clustering problem as a Bayesian model selection problem. Besides a formal search, they also propose heuristics for finding the number of clusters and the sizes of the HMM models, which can overcome the computational complexity of an intensive search. In a simulation experiment the authors conclude that their heuristic methodology works well for clustering sequences with continuous output.

3 Discrete-Output HMM

In this section we briefly describe the methodology of the model underlying the mixture used for sequence clustering in the DBHC algorithm: the HMM with discrete outputs. The HMM is a model for a time series with a finite number of latent states, sometimes also called regimes, which can capture hidden dynamic relations. In the model, emission probabilities can differ between distinct states, states which are assumed to develop according to a first-order Markov process. We describe HMMs where each hidden state defines a categorical distribution over a set of discrete output labels.

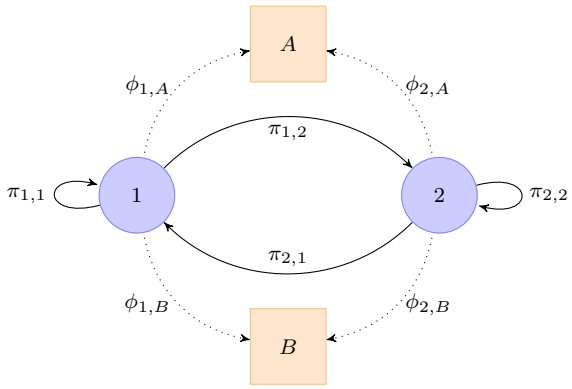

Fig. 1 Visual Representation of an HMM With Two States $\Phi = \{1, 2\}$ and Two Output Labels $\Sigma = \{A, B\}$

3.1 HMM Definitions

Let $\mathbf{X} = (X_1, \ldots, X_T)'$ be a sequence of variables that we observe over a set of discrete time periods $t \in \{1, \ldots, T\}$. In the standard HMM, we consider an unobserved state variable $Z_t \in \Omega$, with $\Omega = \{1, \ldots, H\}$ being the set of states. We use these unobserved states to model the observed output variable X_t. Output variable X_t depends via an emission probability matrix Φ directly on the corresponding unobserved state Z_t, which is assumed to follow a first-order Markov process with some transition matrix Π. Matrix Π is a stochastic matrix that is defined for all pairs of states in $\Omega \times \Omega$. We call Π a right stochastic matrix, i.e., the rows of Π sum to 1. We consider HMMs with a discrete output variable X_t, defined over the labels in Σ. Emission probabilities in stochastic matrix Φ describe the relations between X_t and Z_t, defined for every pair of states and labels in $\Omega \times \Sigma$. Matrix Φ is also a right stochastic matrix, i.e., the rows of Φ sum to 1. The transition probabilities in Π describe the one-period state transitions $\pi_{z,y} = \Pr[Z_t = y | Z_{t-1} = z]$, where $\pi_{z,y}$ is the element in the z-th row and the y-th column of Π. A vector $\boldsymbol{\pi}$ describes the probability distribution for the initial state Z_1. For every time period t we define an emission probability ϕ_{z,A_l} of observing label A_l given the current state Z_t, say z: $\phi_{z,A_l} = \Pr[X_t = A_l | Z_t = z]$. Here ϕ_{z,A_l} is the element in the z-th row and the l-th column of Φ. Figure 1 shows the dependencies of a two-state HMM with a set of two output labels. The total number of free parameters in an HMM is $(H-1) + H(H-1) + H(L-1)$. One should consider the number of free parameters for an HMM, as the probability vector and the rows of the probability matrices sum to 1 and therefore restrict some of the probabilities. For example, the two-state HMM with two labels in Figure 1 has $(2-1)+2(2-1)+2(2-1) = 5$ free parameters.

Parameters Π, Φ, and $\boldsymbol{\pi}$ should be estimated from the data. This is a relatively difficult estimation procedure, as the path of hidden states $Z_1, \ldots, Z_T$ is not observed. We first define a few extra probabilities before we describe the estimation procedure. We call the joint probability that the HMM moves from state z at time $t-1$ to state y at time t and emits x_t the weight of the transition, denoted by $w_{z,y,t-1,x_t}$, $z, y \in \Omega$. The weight then simply equals the product of a transition

probability and an emission probability:

$$\begin{aligned} w_{z,y,t-1,x_t} &= \Pr\left[Z_t = y \,|\, Z_{t-1} = z\right] \cdot \Pr[X_t = x_t \,|\, Z_t = y] \\ &= \pi_{z,y} \cdot \phi_{y,x_t}. \end{aligned} \tag{1}$$

For the zero-th period we define the initial weight as

$$\begin{aligned} w_{0,z,0,x_1} &= \Pr\left[Z_1 = z\right] \cdot \Pr[X_1 = x_1 \,|\, Z_1 = z] \\ &= \pi_z \cdot \phi_{z,x_1}, \end{aligned} \tag{2}$$

where π_z is the z-th element of the vector of initial probabilities $\boldsymbol{\pi}$ and where we denote the non-existing preceding state by 0. For the sake of completeness we also define a weight for the T-th period, called w_T, which equals 1 irrespective of the state at time T because there is no time period $T+1$. That is, the HMM ends with probability 1 at time T. Note that we can now write the joint probability for observations $\mathbf{x} = (x_1, \ldots, x_T)'$ of a sequence $\mathbf{X} = (X_1, \ldots, X_T)'$ and a path $Z_1, \ldots, Z_T$ as a product of weights:

$$\begin{aligned} &\Pr[\mathbf{X} = \mathbf{x}, Z_1 = z_1, \ldots, Z_T = z_T] \\ &\quad = \Pr[\mathbf{X} = \mathbf{x} \,|\, Z_1 = z_1, \ldots, Z_T = z_T] \cdot \Pr[Z_1 = z_1, \ldots, Z_T = z_T] \\ &\quad = \prod_{t=1}^{T} \Pr[X_t = x_t \,|\, Z_t = z_t] \cdot \Pr[Z_t = z_t \,|\, Z_{t-1} = z_{t-1}] \\ &\quad = \prod_{t=1}^{T} \phi_{z_t,x_t} \cdot \pi_{z_{t-1},z_t} \\ &\quad = \prod_{t=1}^{T} w_{z_{t-1},z_t,t-1,x_t}, \end{aligned} \tag{3}$$

using the definitions of weights from (1) and (2). Furthermore, we define forward and backward probabilities to simplify notation, known from the *forward-backward algorithm* for HMMs (Rabiner, 1989). These are recursive probabilities, which can be calculated either by going forward in the sequence of states $Z_1, \ldots, Z_T$ or by going backward in this sequence, after which they are named. Define $forward_{z,t}$ as the forward probability of being in state z at time point t and observing the sequence $x_1, \ldots, x_t$, given weights $w_{z,y,q,x_{q+1}}$, $q \in \{1, \ldots, t-1\}$. We calculate $forward_{z,t}$ recursively as

$$\begin{aligned} &forward_{z,t}(x_1, \ldots, x_t) \\ &\quad = \sum_{y \in \Omega} forward_{y,t-1}(x_1, \ldots, x_{t-1}) \cdot w_{y,z,t-1,x_t}, \ \forall z \in \Omega, \, t = 2, \ldots, T, \end{aligned} \tag{4}$$

using the definitions of weights from (1) and (2). The recursion is initialised by initial weight $w_{0,z,0,x_1}$, i.e., $forward_{z,1}(x_1, \ldots, x_t) = w_{0,z,0,x_1} = \pi_z \cdot \phi_{z,x_1}, \forall z \in \Omega$. In the recursion we sum over all possible states for time periods before t. In a similar fashion, define $backward_{z,t}$ as the backward probability of being in state z

at time point t and observing the sequence $x_{t+1}, \ldots, x_T$, given weights $w_{z,y,q,x_{q+1}}$, $q \in \{t, \ldots, T\}$. We then calculate $backward_{z,t}$ as

$$\begin{aligned} &backward_{z,t}(x_{t+1}, \ldots, x_T) \\ &= \sum_{y \in \Omega} backward_{y,t+1}(x_{t+2}, \ldots, x_T) \cdot w_{z,y,t,x_{t+1}}, \ \forall z \in \Omega, \ t = 1, \ldots, T-1, \end{aligned} \quad (5)$$

again using the definitions of weights in (1) and (2). This recursion is initialised by the weight for the T-th period w_T, i.e., $backward_{z,T} = w_T = 1, \forall z \in \Omega$. We can now break down the probability that the HMM passes through state z at time t while emitting $\mathbf{x}$ as

$$\Pr[Z_t = z, \mathbf{X} = \mathbf{x}] = forward_{z,t}(x_1, \ldots, x_t) \cdot backward_{z,t}(x_{t+1}, \ldots, x_T). \quad (6)$$

Similarly we can break down the probability of making a transition from state z to y from time t to $t+1$ and observing $\mathbf{x}$ as

$$\begin{aligned} \Pr[Z_t = z, Z_{t+1} = y, \mathbf{X} = \mathbf{x}] = &forward_{z,t}(x_1, \ldots, x_t) \\ &\cdot w_{z,y,t,x_t} \cdot backward_{y,t+1}(x_{t+2}, \ldots, x_T). \end{aligned} \quad (7)$$

Now we can also calculate the probability of observing an entire sequence using forward probabilities, which we call $forward(\mathbf{x})$. The expression can then be evaluated as a product of terms as

$$forward(\mathbf{x}) = \Pr[\mathbf{X} = \mathbf{x}] = \prod_{t=1}^{T} \sum_{z \in \Omega} \sum_{y \in \Omega} w_{y,z,t-1,x_t}. \quad (8)$$

We use the forward and backward probabilities to calculate the so-called responsibility profile of the HMM. The responsibility profile consists of the probabilities of passing through a state z and of the probabilities of making the transition from a state y to a state z, all given the observed sequence $\mathbf{x} = (x_1, \ldots, x_T)'$. The first probability of passing through state z at time t equals

$$\begin{aligned} \Pr[Z_t = z \,|\, \mathbf{X} = \mathbf{x}] &= \frac{\Pr[Z_t = z, \mathbf{X} = \mathbf{x}]}{\Pr[\mathbf{X} = \mathbf{x}]} \\ &= \frac{forward_{z,t}(x_1, \ldots, x_t)}{forward(\mathbf{x})} \cdot backward_{z,t}(x_{t+1}, \ldots, x_T), \end{aligned} \quad (9)$$

where $forward(\mathbf{x})$ is the forward probability of observing the entire sequence $\mathbf{x}$. The second probability of making the transition from state y to state z between time points t and $t+1$ equals

$$\begin{aligned} \Pr[Z_t = z, Z_{t+1} = y \,|\, \mathbf{X} = \mathbf{x}] &= \frac{\Pr[Z_t = z, Z_{t+1} = y, \mathbf{X} = \mathbf{x}]}{\Pr[\mathbf{X} = \mathbf{x}]} \\ &= \frac{forward_{z,t}(x_1, \ldots, x_t)}{forward(\mathbf{x})} \cdot w_{z,y,t,x_{t+1}} \cdot backward_{y,t+1}(x_{t+2}, \ldots, x_T), \end{aligned} \quad (10)$$

which we again define in terms of forward and backward probabilities, combined with the forward probability of observing the entire sequence $\mathbf{x}$. We use the probabilities in (9) and (10) for estimating HMM parameters with the *Baum-Welch* algorithm, which we describe in the next paragraph.

3.2 Baum-Welch Learning

The Baum-Welch algorithm is a special case of the expectation-maximisation (EM) algorithm of Dempster et al. (1977), specifically designed for obtaining parameters in an HMM (Rabiner, 1989). The algorithm alternates between estimating a responsibility profile given the current HMM parameters in the E-step, and maximising the likelihood by updating the HMM parameters given the current responsibility profile in the M-step. The steps are alternated until the likelihood of the HMM converges. In the E-step we calculate the responsibility profile using equations (9) and (10). The probabilities in the responsibility profile can be seen as expectations of the path of hidden states that has generated the observed data. For the M-step we define expressions $\mathrm{T}^t_{z,y}$, $\mathrm{E}^t_z(A_l)$, and I_z based on the probabilities in the responsibility profile:

$$\begin{aligned}
\mathrm{T}^t_{z,y} &= \Pr\left[Z_t = z, Z_{t+1} = y \,|\, \mathbf{X} = \mathbf{x}\right], \quad t = 1, \ldots, T-1 \\
\mathrm{E}^t_z(A_l) &= \begin{cases} \Pr\left[Z_t = z \,|\, \mathbf{X} = \mathbf{x}\right] & \text{if } x_t = A_l \\ 0 & \text{otherwise} \end{cases}, \quad t = 1, \ldots, T \\
\mathrm{I}_z &= \Pr\left[Z_1 = z \,|\, \mathbf{X} = \mathbf{x}\right], \qquad (11)
\end{aligned}$$

where $\mathrm{T}^t_{z,y}$ is a transition probability at time t, $\mathrm{E}^t_z(A_l)$ an emission probability at time t, and I_z an initial probability. We summarise these probabilities in (11) into estimates of probabilities for an entire sequence by summing them over the relevant time periods and scaling this sum with respect to all other possibilities for either transition or emission. We update the parameter estimates of emission and transition probabilities in the current iteration, say m, based on the responsibility profile as follows:

$$\begin{aligned}
\hat{\pi}^{(m)}_{z,y} &= \frac{\sum_{t=1}^{T-1} \mathrm{T}^t_{z,y}}{\sum_{t=1}^{T-1} \sum_{q \in \Omega} \mathrm{T}^t_{z,q}} \\
\hat{\phi}^{(m)}_{z,A_l} &= \frac{\sum_{t=1}^{T} \mathrm{E}^t_z(A_l)}{\sum_{t=1}^{T} \sum_{h \in \Sigma} \mathrm{E}^t_z(A_h)}, \qquad (12)
\end{aligned}$$

where the responsibility profile in iteration m is calculated using the parameter estimates from iteration $m-1$. Estimates for the initial state probabilities in $\boldsymbol{\pi}$ are simply:

$$\hat{\pi}^{(m)}_z = \frac{\mathrm{I}_z}{\sum_{q \in \Omega} \mathrm{I}_q}. \qquad (13)$$

These parameter estimates maximise the likelihood of an HMM in (14), given the current expectations of the path of hidden states—i.e., the responsibility profile (Rabiner, 1989). The likelihood is simply the probability that the HMM emitted the sequence $\mathbf{x}$ given the estimated parameters. Let φ denote the collection of the HMM parameters π, $\mathit{\Pi}$, and $\mathit{\Phi}$ for the sake of brevity, $\varphi = (\pi, \mathit{\Pi}, \mathit{\Phi})$. We evaluate the likelihood of the m-th iteration in the algorithm as follows:

$$\begin{aligned}
L^{(m)}(\varphi \,|\, \mathbf{X}) &= \Pr\left[\mathbf{X} = \mathbf{x} \,\middle|\, \hat{\varphi}^{(m)}\right] \\
&= \prod_{t=1}^{T} \sum_{z \in \Omega} \sum_{y \in \Omega} w_{y,z,t-1,x_t} \\
&= forward(\mathbf{x}), \qquad (14)
\end{aligned}$$

Initialise HMM parameter estimates $\hat{\varphi}^{(0)} \leftarrow \varphi_0$
Initialise $m \leftarrow 1$
repeat
  E-step: Update responsibility profile in (9) and (10) given estimates in $\hat{\varphi}^{(m-1)}$
  M-step: Maximise $L^{(m)}(\varphi \,|\, \mathbf{X})$ in (14) by updating $\hat{\varphi}^{(m)}$ in (12) and (13) given current responsibility profile
  Calculate likelihood $L^{(m)}(\varphi \,|\, \mathbf{X})$ using current parameter estimates $\hat{\varphi}^{(m)}$
  Update $m \leftarrow m + 1$
until $\left|L^{(m)}(\varphi \,|\, \mathbf{X}) - L^{(m-1)}(\varphi \,|\, \mathbf{X})\right| < \varepsilon_1$
Accept estimates in $\hat{\varphi}^{(m)}$ as parameters of the HMM

Fig. 2 Baum-Welch Algorithm

where we calculate $forward(\mathbf{x})$ using the estimated parameters of the m-th iteration $\hat{\varphi}^{(m)}$. The algorithm converges when the likelihoods in two consecutive iterations differ less than some small number ε_1. The algorithm should be initialised with some initial guesses for the parameters in φ, $\varphi_0 = (\boldsymbol{\pi}_0, \mathit{\Pi}_0, \mathit{\Phi}_0)$. Usually, these guesses are obtained by random draws from Dirichlet distributions: $\boldsymbol{\pi}_0 \sim \mathrm{Dir}(1, \mathbf{1}_H)$, $\mathit{\Pi}_0 \sim \mathrm{Dir}(H, \mathbf{1}_H)$, and $\mathit{\Phi}_0 \sim \mathrm{Dir}(H, \mathbf{1}_L)$, where $\mathbf{1}_q$ is a $q \times 1$ vector of ones and $\mathrm{Dir}(n, \boldsymbol{\alpha})$ is a Dirichlet distribution with dimensionality n and hyper parameter vector $\boldsymbol{\alpha}$. Figure 2 denotes the pseudocode for the Baum-Welch algorithm.

4 Sequence Clustering with the DBHC Algorithm

In this section we describe clustering using HMMs with the DBHC algorithm. The DBHC algorithm is based on the Bayesian HMM Clustering algorithm of Li and Biswas (2000), which is intended for sequences with continuous observations in discrete time, while the DBHC algorithm is intended for sequences with discrete observations in discrete time. The clustering of sequences using HMM representations was first mentioned in Rabiner et al. (1989). The idea behind HMM Clustering is to model a different HMM per cluster, that is, modelling a mixture of K HMMs. Let the observed output variable be $X_{i,t}$, gathered in a sequence $\mathbf{X}_i$ for individual i. Let the mixture distribution be described by a discrete variable S, where a value k of S represents a cluster. The probability that sequence i belongs to cluster k is denoted by $p_{i,k}$. The probability of observing sequence $\mathbf{x}_i$ is then modelled as follows:

$$\Pr\left[\mathbf{X}_i = \mathbf{x}_i \,|\, \varphi\right] = \sum_{k=1}^{K} p_{i,k} \Pr\left[\mathbf{X}_i = \mathbf{x}_i \,|\, S_i = k; \varphi_k\right], \tag{15}$$

where we model $\Pr\left[\mathbf{X}_i = \mathbf{x}_i \,|\, S_i = k; \varphi_k\right]$ with an HMM. Parameter φ_k denotes the HMM parameters of cluster k, and φ the collection of all φ_k, $\varphi = (\varphi_1, \ldots, \varphi_K)$.

4.1 Learning an HMM from Multiple Sequences

A cluster can contain multiple individuals, therefore we need to be able to train a single HMM on multiple observed sequences. Training an HMM on multiple sequences is very similar to training an HMM on a single sequence if we assume that

the sequences are generated independently from each other by the HMM (Rabiner, 1989). In the E-step of the Baum-Welch algorithm, we update the responsibility profile of each sequence in a cluster using the current parameters of the HMM for that cluster. We perform this E-step for all sequences in all clusters. In the M-step we then adjust the HMM parameters of each cluster using the responsibility profiles of all sequences in each cluster separately. That is, we train the HMM of a cluster on all sequences in the cluster, i.e., we train the HMM on multiple sequences. For this purpose we define the responsibility profile of sequence i in cluster k as follows:

$$\begin{aligned} &\Pr\left[Z_{i,t} = z \,|\, \mathbf{X}_i = \mathbf{x}_i; \hat{\varphi}_k\right] \\ &\Pr\left[Z_{i,t} = z, Z_{i,t+1} = y \,|\, \mathbf{X}_i = \mathbf{x}_i; \hat{\varphi}_k\right], \end{aligned} \tag{16}$$

which we calculate in the same way as (9) and (10) using the estimated HMM parameters in $\hat{\varphi}_k$. We also define the probabilities in (11) for each individual i in cluster k:

$$\begin{aligned} \mathrm{T}^t_{i,z,y} &= \Pr\left[Z_{i,t} = z, Z_{i,t+1} = y \,|\, \mathbf{X}_i = \mathbf{x}_i; \hat{\varphi}_k\right], \quad t = 1, \ldots, T-1 \\ \mathrm{E}^t_{i,z}(A_l) &= \begin{cases} \Pr\left[Z_{i,t} = z \,|\, \mathbf{X}_i = \mathbf{x}_i; \hat{\varphi}_k\right] & \text{if } x_{i,t} = A_l \\ 0 & \text{otherwise} \end{cases}, \quad t = 1, \ldots, T \\ \mathrm{I}_{i,z} &= \Pr\left[Z_{i,1} = z \,|\, \mathbf{X}_i = \mathbf{x}_i; \hat{\varphi}_k\right], \end{aligned} \tag{17}$$

where $\mathrm{T}^t_{i,z,y}$ is a transition probability at time t, $\mathrm{E}^t_{i,z}(A_l)$ an emission probability at time t, and $\mathrm{I}_{i,z}$ an initial probability, all for the sequence of individual i. Instead of maximising the likelihood of a single sequence, we now update the parameter estimates in iteration m by maximising the joint likelihood of observing the sequences in cluster k given the responsibility profile in iteration m, i.e., the parameters from the previous iteration $m-1$:

$$\begin{aligned} &\Pr\left[\mathbf{X}_1 = \mathbf{x}_1, \ldots, \mathbf{X}_{N_k} = \mathbf{x}_{N_k} \,\middle|\, \hat{\varphi}_k^{(m-1)}\right] \\ &\quad = \prod_{i=1}^{N_k} \Pr\left[\mathbf{X}_i = \mathbf{x}_i \,\middle|\, \hat{\varphi}_k^{(m-1)}\right] \\ &\quad = \prod_{i=1}^{N_k} forward(\mathbf{x}_i), \end{aligned} \tag{18}$$

where we use the assumption that the sequences are independently generated to split up the likelihood and where N_k is the number of individuals in cluster k. This means that we can maximise the likelihoods of the sequences independently. Rabiner (1989) then shows that we can update the parameter estimates for cluster k in iteration m in a similar way as in (12) and (13), since the probabilities in the responsibility profile are scaled by the forward probabilities of observing the

respective sequence:

$$\hat{\pi}^{(m)}_{k,z,y} = \frac{\sum_{i=1}^{N_k} \sum_{t=1}^{T-1} \mathrm{T}^t_{i,z,y}}{\sum_{i=1}^{N_k} \sum_{t=1}^{T-1} \sum_{q \in \Omega_k} \mathrm{T}^t_{i,z,q}}$$
$$\hat{\phi}^{(m)}_{k,z,A_l} = \frac{\sum_{i=1}^{N_k} \sum_{t=1}^{T} \mathrm{E}^t_{i,z}(A_l)}{\sum_{i=1}^{N_k} \sum_{t=1}^{T} \sum_{m \in \Sigma} \mathrm{E}^t_{i,z}(A_m)}$$
$$\hat{\pi}^{(m)}_{k,z} = \frac{\sum_{i=1}^{N_k} \mathrm{I}_{i,z}}{\sum_{i=1}^{N_k} \sum_{q \in \Omega_k} \mathrm{I}_{i,q}}. \tag{19}$$

We use the Baum-Welch algorithm for multiple sequences in our clustering algorithm. Clustering with HMMs adds an extra dimension to the clustering problem, namely that for each cluster not only the number of clusters K for the traditional clustering problem needs to be determined, but also the number of states for each HMM. Following a Bayesian approach, Li and Biswas (2000) incorporate this search in the parameter inference.

4.2 DBHC Algorithm

Li and Biswas (2000) propose a Bayesian estimation method that incorporates both the search for the optimal cluster partition and the search for the number of states per cluster. They call the latter search the model size selection problem. We adapt their strategy to a setting with discrete output. In the Bayesian framework, both the clustering problem and the model size selection problem can be seen as Bayesian model selection problems. That is, selecting the model M with the highest posterior probability: $\Pr[M \mid X]$. If we were to compare two models M_1 and M_2, we would consider the ratio of their posterior probabilities:

$$\frac{\Pr[M_2 \mid X]}{\Pr[M_1 \mid X]} = \frac{\frac{\Pr[M_2]\Pr[X \mid M_2]}{\Pr[X]}}{\frac{\Pr[M_1]\Pr[X \mid M_1]}{\Pr[X]}} = \frac{\Pr[M_2]\Pr[X \mid M_2]}{\Pr[M_1]\Pr[X \mid M_1]}, \tag{20}$$

where we use Bayes' rule to develop the posteriors. If we do not favor any model in advance, that is, if we do not favor any number of clusters or any model size for the HMMs, we have $\Pr[M_1] = \Pr[M_2]$. The ratio of model posteriors in (20) would then simplify to the ratio of likelihoods:

$$\frac{\Pr[M_2 \mid X]}{\Pr[M_1 \mid X]} = \frac{\Pr[X \mid M_2]}{\Pr[X \mid M_1]}. \tag{21}$$

In conclusion, when choosing between different models we can simply select the model with the largest likelihood. In case of unobserved variables, this should be the complete data likelihood. We use this strategy for both the search for the number of clusters and for the HMM model size. The complete data likelihoods can be obtained using Markov Chain Monte Carlo methods in case of latent variables, for example with Gibbs sampling (Geman and Geman, 1984).

When using Markov Chain Monte Carlo methods for Bayesian inference, the two search dimensions can turn out to be very costly in terms of computational complexity, namely exponential complexity (Li and Biswas, 2000). The authors therefore propose to use approximations of the log complete data likelihood using the Bayesian Information Criterion (BIC). We use these approximations for both search dimensions. The Bayesian HMM Clustering algorithm uses hard assignment of clusters, that is, $p_{i,k} = 1$ for one value of k, and $p_{i,l} = 0$ for all $l \neq k$. In general the BIC of a model is defined as $\mathrm{BIC} = -2 \log L + d \log N$, where L is the likelihood of the model without unobserved variables, d is the number of parameters, and N is the number of observations.

We choose the number of clusters K such that the BIC for the entire mixture is minimised. Following the approach in Li and Biswas (2000), the number of clusters K is added as a penalty to the BIC, similar to the penalty of the number of parameters. The BIC for a model M^K with K clusters can be calculated as follows:

$$\mathrm{BIC}_K = -2 \sum_{i=1}^{N} \log \left[\sum_{k=1}^{K} p_{i,k} \Pr\left[\mathbf{X}_i \mid S_i = k; \hat{\varphi}_k\right] \right] + \left(K + \sum_{k=1}^{K} d_k \right) \log N, \quad (22)$$

where d_k is equal to the number of free parameters in $\hat{\varphi}_k$ larger than some threshold ε_2, again following the original algorithm. In this way, the BIC only penalises the likelihood for probabilities that are set to a non-zero value larger than ε_2. Note that $p_{i,k} = 1$ if $S_i = k$ and 0 otherwise. Recall that the number of free parameters in an HMM for cluster k is $(H_k - 1) + H_k(H_k - 1) + H_k(L - 1)$. We obtain d_k from the previous number by subtracting the number of parameters that are smaller than ε_2 from the total number of free parameters. Now d_k is a penalty for the number of non-zero parameters. Minimising BIC_K is approximately equivalent to maximising the log complete data likelihood (Li and Biswas, 2000). Note that this is the complete data likelihood as it assumes the cluster memberships known. We choose the number of clusters K such that its corresponding model minimises BIC_K. We start the search with $K = 1$, and iteratively keep increasing K with steps of 1, until BIC_K does not decrease anymore, as suggested by Li and Biswas (2000).

For each cluster we determine the number of HMM states in a likewise fashion. In a similar way as shown in (20) and (21), we can show that we can determine the optimal size H_k for the model of cluster k by maximising the likelihood over a set of possible values for H_k. Again, we approximate the likelihood with the BIC:

$$\mathrm{BIC}_{H_k} = -2 \sum_{i=1}^{N_k} \log \Pr\left[\mathbf{X}_i \mid H_k, \hat{\varphi}_k\right] + d_k \log N_k, \quad (23)$$

where d_k is equal to the number of free parameters in $\hat{\varphi}_k$ larger than some threshold ε_2. We initialise each HMM with $H_k = 1$, and iteratively keep increasing H_k with steps of 1, until BIC_{H_k} does not decrease anymore. This set-up is similar to the search for the optimal number of clusters. Figure 3 denotes the pseudocode for the HMM model size search algorithm for cluster k.

For a given model size, we obtain model parameters for each cluster k with the Baum-Welch algorithm for multiple sequences. For assigning sequences to clusters we use the sequence-to-HMM likelihood measure (Rabiner, 1989). An observed

// Initialisation
Initialise $m \leftarrow 1$
Initialise $H_k \leftarrow 1$
Obtain HMM parameters for an HMM with size H_k using Baum-Welch algorithm
Set $\text{BIC}^{(1)} \leftarrow \text{BIC}_1$
// Model size expansion
repeat
  Update $m \leftarrow m + 1$
  Update $H_k \leftarrow H_k + 1$
  Obtain HMM parameters for an HMM with size H_k using Baum-Welch algorithm
  Set $\text{BIC}^{(m)} \leftarrow \text{BIC}_{H_k}$
until $\text{BIC}^{(m)} \geq \text{BIC}^{(m-1)}$
// Accept final model
Accept model size of iteration $m - 1$

Fig. 3 HMM Model Size Search Algorithm for a Given Cluster k

sequence $\mathbf{x}_i$ is assigned to the cluster k that has the highest likelihood: $\Pr[\mathbf{x}_i \mid \hat{\varphi}_k]$. The sequence-to-HMM likelihood is simply the forward probability of observing that sequence in the HMM of cluster k, i.e., $\Pr[\mathbf{x}_i \mid \hat{\varphi}_k] = forward(\mathbf{x}_i)$. If after all assignments at least one of the sequences switched clusters, we re-estimate the HMM of each cluster without changing the model size, and we redistribute the observations over the new clusters. The process is repeated until no sequence changes clusters after redistribution. Finally, the size search algorithm is invoked for all clusters to make the models best reflect the data in the clusters.

At the start of every iteration in the search for the optimal clustering we have to initialise all models in the partition with a set of observations, a seed. The seed is used to determine the number of HMM states for a cluster before other observations are added. When the size search algorithm is invoked for heterogeneous data—i.e., it contains observations from different true clusters—additional states might be added to reflect the heterogeneity, while we actually want to incorporate this heterogeneity in the cluster memberships. The seed selection procedure aims at preventing this. In each iteration, we rebuild the model for each cluster to reflect the data best according to the number of clusters at that point in the algorithm. A seed consists of r observations, with r usually an arbitrarily chosen small number. For the first seed in every iteration we randomly select a sequence from the set of all sequences. We then estimate an HMM for this observation—with 2 states, as this is just a temporary HMM—and we add the remaining $r - 1$ observations that are best represented by this HMM in terms of sequence-to-HMM likelihood to the seed. For all the other $K - 1$ cluster seeds we select the first sequence as the one that is worst represented by the current set of seed models. We find the remaining $r - 1$ sequences in the same way as for the first seed. In total we now have included r observations in each of the K seeds. Li and Biswas (2000) use $r = 3$ for initialising the models throughout their paper. After the observations for the seed have been selected, we determine the seed model using the size search algorithm in Figure 3.

In the seed selection procedure the likelihood of an HMM is many times developed for sequences the HMM has not been trained on. It is therefore possible that these sequences contain labels the training set for the HMM did not contain—these emissions were assigned probability zero in the Baum-Welch algorithm. The new sequences potentially also require emission-state combinations with zero probabil-

ity in the HMM. All these cases will result in a likelihood value of 0, corresponding to a log likelihood value of $-\infty$. The step that selects the worst represented sequence will then be inconclusive among all sequences with log likelihood $-\infty$, and proceed with a random draw from these sequences. We overcome this inconclusiveness by adding a probability smoothing step to the emission probability matrix after parameter estimation.

Probability smoothing is replacing zero probabilities with very small values to prevent the likelihood of unseen data to be zero, while preserving the condition that all probabilities sum to 1. After parameter estimation with the Baum-Welch algorithm we use the probability smoothing technique of absolute discounting, which has been shown to work well for HMMs in word recognition (Taghva et al., 2005). Absolute discounting is subtracting a small amount of probability from all non-zero probabilities, and dividing this portion equally over the zero probabilities. We apply the smoothing row-wise for the emission probability matrix after parameter estimation.

For a certain state z in the HMM of a cluster k, let the number of non-zero emission probabilities be v. Recall that L is the total number of labels in the alphabet, thus $L - v$ is the number of zero emission probabilities. We subtract a small amount q from each non-zero emission probability, which we then divide equally over the $L-v$ non-zero probabilities. The smoothed probability of ϕ_{k,z,A_l}, say $\widetilde{\phi}_{k,z,A_l}$, then equals:

$$\widetilde{\phi}_{k,z,A_l} = \begin{cases} \phi_{k,z,A_l} - q & \text{if } \phi_{k,z,A_l} > 0 \\ \frac{vq}{L-v} & \text{otherwise} \end{cases}, \quad l = 1, \ldots, L. \tag{24}$$

There is no standard way for determining the smoothing parameter q, but for our algorithm it is important to pick it such that $\frac{vq}{L-v}$ is smaller than ε_2. In this way the smoothed zero probabilities do not count towards the total of number of non-zero parameters d_k.

Now we present the entire DBHC algorithm with the order in which we execute the search and estimation steps, including the probability smoothing step for discrete sequence data. Figure 4 denotes the pseudocode for the algorithm. As described in Section 3.2, the Baum-Welch algorithm requires starting values for the HMM parameters. In the HMM size search algorithm we build the HMMs with random starting values for the parameters as we do not have any information on the sequences in a cluster yet. For building temporary HMMs—used for finding either the most similar or the most dissimilar sequences in the seed selection procedure—we use random starting values for the same reason. However, during the updating of HMM parameters in the object redistribution phase at the end of each iteration, we initialise the HMM for a cluster with the parameters that were estimated for the cluster before the most recent redistribution. We do this in order to facilitate convergence during the alternation between object redistribution and parameter updating. Whenever we do initialise an HMM with random starting values for the parameters, we use 10 of these random starts and select the model that achieves the highest likelihood among these, following Helske and Helske (2017). We use only 10 random starts as this estimation procedure is often repeated in the algorithm and higher numbers of random starts would lay an extra burden of computational complexity on the algorithm.

Finally, the DBHC algorithm is implemented in the **DBHC** package of the R statistical software (XXX (blinded for review), 2019). Default hyperparameter val-

// Initialisation
Initialise $m \leftarrow 1$ // iteration
Initialise $K \leftarrow 1$ // number of clusters
Set seed size r // seed size
// First iteration
Select random first observation from $1, \ldots, N$
Build temporary HMM with 2 states based on first observation
Smooth emission probabilities of temporary HMM
Evaluate sequence-to-HMM likelihood based on temporary HMM for all sequences not yet selected
Add $r - 1$ observations to seed with highest sequence-to-HMM likelihood
Apply HMM model size search in Figure 3 to first cluster
Smooth emission probabilities of HMM
Add all sequences to first cluster
Set $\mathrm{BIC}^{(1)} \leftarrow \mathrm{BIC}_1$ first clustering
// Partition expansion
repeat
  Update $m \leftarrow m + 1$
  Update $K \leftarrow K + 1$
  // Seed selection
  for $i \leftarrow 1$, K **do**
    if i = 1 **then**
      Select random first observation from $1, \ldots, N$
    else
      Evaluate sequence-to-HMM likelihood for every seed model in iteration i for all sequences not yet selected
      Consider for each sequence the HMM for which it has the highest sequence-to-HMM likelihood
      Select first observation as the one with lowest such sequence-to-HMM likelihood: the sequence worst represented by the current seed models
    end if
    Build temporary HMM with 2 states based on first observation
    Smooth emission probabilities of temporary HMM
    Evaluate sequence-to-HMM likelihood based on temporary HMM for sequences not yet selected
    Add $r - 1$ observations to seed with highest sequence-to-HMM likelihood
    Apply HMM model size search in Figure 3 to the seed
    Smooth emission probabilities of HMM
  end for
  Set seed models as the cluster models
  // Object redistribution
  Assign each sequence to the cluster model for which it has highest sequence-to-HMM likelihood
  repeat
    Update HMM parameters with Baum-Welch algorithm in Figure 2 for all clusters
    Smooth emission probabilities of each HMM
    Assign each sequence to the cluster model for which it has highest sequence-to-HMM likelihood
  until No sequence switches between clusters
  Accept current sequence distribution
  Set $\mathrm{BIC}^{(m)} \leftarrow \mathrm{BIC}_K$ current clustering
until $\mathrm{BIC}^{(m)} \geq \mathrm{BIC}^{(m-1)}$
// Accept final model
Accept clustering in iteration $m - 1$ as the model

Fig. 4 DBHC algorithm

ues in the software package correspond to the values suggested in this paragraph. For example, the default number of hidden states in an initial HMM is set equal

Table 1 Sequence Labels Appearing in Swiss Household Data

Original	Label	Family status
0	P	Living with parents
1	L	Left home
2	M	Married
3	LM	Left home, married
4	C	Having children
5	LC	Left home, having children
6	LMC	Left home, married, having children
7	D	Divorced

Table 2 Clustering Results Swiss Household Data

Cluster	# Individuals	# Hidden States
1	157	2
2	66	3
3	211	3
4	425	6
5	298	4
6	843	5

to 2 and the default number of observations in a seed is set equal to 3. These and other hyperparameters can be controlled by the user of the software package, see XXX (blinded for review) (2019). For the analysis in Section 5 we use utilise this R implementation of the algorithm.

5 Illustrative Example

For illustrating how one can perform sequence clustering using the DBHC algorithm, we perform an analysis on the `biofam` data set from R package **TraMineR**. These data consist of family life sequences built from a biographical survey carried out by the Swiss Household Panel (SHP) in 2002 (Gabadinho et al., 2011). The data contains 16-year-long sequences with yearly observations of family status for 2000 individuals who were at least 30 years old at the time of the survey. Table 1 shows the labels appearing in the data set, that is, the statuses recorded in the survey for the 2000 individuals, along with the original labels attached to these statuses in `biofam`. We replace the non-intuitive labels 0 to 7—those that appear in the original data set—with the more intuitive ones in Table 1.

Table 2 shows descriptives of the results found by the DBHC algorithm for the Swiss household data. We have run the algorithm with the default hyperparameter values described in Section 4.2, e.g., we set the seed size equal to 3. The algorithm finds 6 clusters for the `biofam` data set, with the number of hidden states ranging from 2 to 6. The cluster sizes range from 66 individuals in cluster 2 to roughly 800 individuals in cluster 6. Here, an 'individual' of course corresponds to a sequence belonging to an individual.

To illustrate how we can interpret clusters using the hidden states of the cluster model, we study the heatmaps of the HMM of one of the clusters, say cluster 5, which contains roughly 300 individuals and for which the algorithm finds 4 hidden

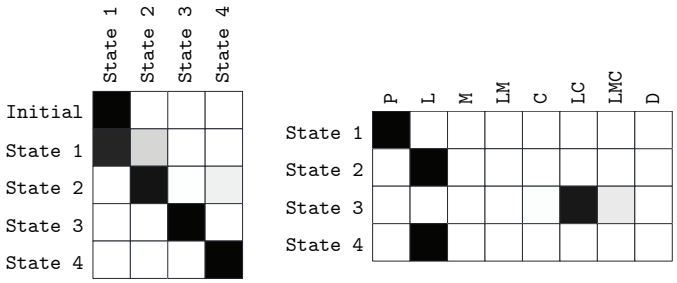

Fig. 5 Heatmaps of initial and transition probability matrix (left) and emission probability matrix (right) of cluster 5

states. Figure 5 shows heatmaps of the initial, transition, and emission probability matrices of this cluster. We can see that people in this cluster initially start in state 1 and—looking at the emission probability matrix—they are living with their parents in this state. After being in this state 1, there is a small probability to move to state 2, where they leave home—although the majority stays in state 1. We observe that after state 2, part of the individuals moves to state 4, where they remain in the status 'left home'. Individuals in state 4 tend to stay in state 4. This leaves us with state 3, for which it is not visible by which state it is preceded in the heatmap, i.e., apparently the probability of moving to state 3 is marginal. Note that the corresponding emissions of state 3 therefore also rarely occur for this cluster. This leaves us to conclude that cluster 5 consists of individuals who start out living with their parents, after which there is a probability of transitioning into the state of leaving home, where they will stay until the end once this has occurred, without getting married or having children. A random sample of 5 sequences drawn from the ones contained in this cluster confirms this intuition:

```
1: P-P-P-P-P-P-L-L-L-L-L-L-L-L-L
2: P-P-P-P-P-P-P-P-P-P-L-L-L-L-L
3: P-P-L-L-L-L-L-L-L-L-L-L-L-L-L
4: P-P-P-P-P-P-L-L-L-L-L-L-L-L-L
5: P-P-L-L-L-L-L-L-L-L-L-L-L-L-L.
```

Similar analyses can be carried out for the remaining cluster models, which one would use to obtain a complete view of the partition found by the DBHC algorithm. Doing this, we obtain descriptions of the other clusters too. Individuals in cluster 1 live with their parents and stay there for the entire observation period. In cluster 2, individuals start out already having left home and marry eventually, combined with either directly having children or having children at some later point in time. Individuals in cluster 3 start out living with their parents and go into marriage without leaving their paternal home or having children. In cluster 4, individuals start out living with their parents, after which they leave home and later get married with a potential for having children, a divorce, or a combination of those two. Cluster 6 is the biggest cluster with 843 individuals, for which the algorithm finds 5 hidden states. Individuals in cluster 6 start out living with their parents and then follow all kinds of different paths before they eventually end

up having left home, being married, and having children. With this cluster we can conclude that it is the most common path to start out living with parents and eventually end up moving out, being married and having children. Note that the path identified for cluster 4 is the only one that involves the potential for a divorce. In total, the algorithm identifies roughly 6 different life paths, each path corresponding to one of the clusters. In this way, we have illustrated how hidden states can help interpreting cluster memberships in sequence data sets.

6 Conclusion

We have presented the DBHC algorithm for sequence clustering using HMMs and we have shown how the hidden states in a mixture of HMMs can aid the interpretation task of a cluster analysis for sequence data. The DBHC algorithm extends the existing Bayesian HMM Clustering algorithm to be applicable to discrete sequence data by adding a probability smoothing step with absolute discounting to its seed selection procedure. The DBHC algorithm is completely self-contained as it incorporates both the search for the number of clusters and the search for the number of hidden states in each cluster model, next to the parameter estimation procedure with the Baum-Welch algorithm for each cluster model. The algorithm can be used to obtain a mixture of HMMs for discrete sequence data and is implemented in the **DBHC** package of the R statistical software. We can conclude that the algorithm works well as it provides well-interpretable clusters for our application. In future work, we would like to explore whether a full Bayesian approach—as opposed to the current semi-Bayesian approach—improves the fit of the mixture model. Unfortunately, this will most likely put an extra computational burden on the algorithm complexity and research into the feasibility is required.

References

Burke J, Davison D, Hide W (1999) d2_cluster: a validated method for clustering EST and full-length cDNA sequences. *Genome Research* 9(11):1135–1142

Cadez I, Heckerman D, Meek C, Smyth P, White S (2003) Model-based clustering and visualisation of navigation patterns on a web site. *Data Mining and Knowledge Discovery* 7(4):399–424

Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B* pp 1–38

Dong G, Pei J (2007) *Sequence Data Mining*. Springer Science & Business Media

Gabadinho A, Ritschard G, Müller NS, Studer M (2011) Analysing and visualising state sequences in R with TraMineR. *Journal of Statistical Software* 40(4):1–37

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

Helske J, Helske S (2017) Mixture Hidden Markov Models for Sequence Data: The seqHMM Package in R. URL `https://cran.r-project.org/package=seqHMM`, R package version 1.0.8

Li C, Biswas G (2000) A Bayesian approach to temporal data clustering using hidden Markov models. In: *Proceedings of the 17th International Conference on Machine Learning*, Morgan Kaufmann Publishers Inc., pp 543–550

R Core Team (2017) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, URL `https://www.R-project.org/`

Rabiner LR (1989) A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2):257–286

Rabiner LR, Lee CH, Juang BH, Wilpon JG (1989) HMM clustering for connected word recognition. In: *1989 International Conference on Acoustics, Speech, and Signal Processing*, IEEE, pp 405–408

Smyth P (1997) Clustering sequences with hidden Markov models. In: *Advances in Neural Information Processing Systems*, MIT Press, pp 648–654

Stolcke A, Omohundro SM (1994) Best-first model merging for hidden Markov model induction. *CoRR* URL `http://arxiv.org/abs/cmp-lg/9405017`, `cmp-lg/9405017`

Taghva K, Coombs JS, Pereda R, Nartker TA (2005) Address extraction using hidden markov models. *Proc SPIE 5676, Document Recognition and Retrieval XII* pp 119–126

XXX (blinded for review) (2019) DBHC: Sequence Clustering with Discrete-Output HMMs. URL `https://CRAN.R-project.org/package=DBHC`, R package version 0.0.2