Bayesian Analysis (2016) **11**, Number 1, pp. 247–263

Recursive Learning for Sparse Markov Models

Jie Xiong*, Väinö Jääskinen†, and Jukka Corander‡,§

Abstract. Markov chains of higher order are popular models for a wide variety of applications in natural language and DNA sequence processing. However, since the number of parameters grows exponentially with the order of a Markov chain, several alternative model classes have been proposed that allow for stability and higher rate of data compression. The common notion to these models is that they cluster the possible sample paths used to predict the next state into invariance classes with identical conditional distributions assigned to the same class. The models vary in particular with respect to constraints imposed on legitime partitions of the sample paths. Here we consider the class of sparse Markov chains for which the partition is left unconstrained *a priori*. A recursive computation scheme based on Delaunay triangulation of the parameter space is introduced to enable fast approximation of the posterior mode partition. Comparisons with stochastic optimization, k-means and nearest neighbor algorithms show that our approach is both considerably faster and leads on average to a more accurate estimate of the underlying partition. We show additionally that the criterion used in the recursive steps for comparison of triangulation cell contents leads to consistent estimation of the local structure in the sparse Markov model.

Keywords: clustering, Delaunay triangulation, recursive learning, sequence analysis, sparse Markov chains.

1 Introduction

Markov chains of higher order are useful models for capturing spatial or time-dependence in sequential data where the dependences extend beyond immediate neighbors. However, the rapidly increasing parameter richness is a challenge for such models and, therefore, several sparser alternatives have been considered, in particular in the machine learning literature. Finite memory sources and variable order Markov chain models are popular methods in data compression and mining for applications in bioinformatics and natural language modeling. A shared feature of these models is that they cluster the possible sample paths used to predict the next state into invariance classes, where identical conditional distributions are assigned to the same class. The main difference between the models arises from the constraints imposed on legitime partitions of the sample paths.

Here we consider estimation in the class of sparse Markov chains (SMC) for which the partition is left unconstrained *a priori* (Jääskinen et al., 2013). Such models need not correspond to a hierarchical representation of contexts used in variable length Markov chains (Rissanen et al., 1983; Bühlmann et al., 1999) and can lead to significantly improved predictions and higher rate of data compression than variable length Markov

*Department of Mathematics and Statistics, University of Helsinki, jie.xiong@helsinki.fi
†Department of Mathematics and Statistics, University of Helsinki, vaino.jaskinen@helsinki.fi
‡Department of Mathematics and Statistics, University of Helsinki, jukka.corander@helsinki.fi
§Department of Mathematics, Åbo Akademi University

 DOI: 10.1214/15-BA949

chains and ordinary Markov chains. Overall, very limited experience is currently available about learning algorithms for sparse models of this type. Gonzalez-Lopez et al. (2010) derived an asymptotic criterion for learning of partitions for a related class of Markov models, and Farcomeni (2011) considered hidden Markov partition models and proposed learning by the expectation-maximization algorithm (EM), where the number of hidden states is fixed.

Jääskinen et al. (2013) derived a stochastic greedy algorithm for learning SMC models by Bayesian clustering with merge/split and re-allocation operators that are not purely random, in contrast to similar proposals used in standard Markov chain Monte Carlo (MCMC) samplers for clustering. However, as the order of the Markov chain increases, the search space for the algorithm grows exponentially and, consequently, the stochastic greedy optimization method may experience slow convergence. Inspired by the deterministic clustering algorithm introduced in Dahl et al. (2009), we develop here a recursive algorithm for optimizing the partition for an SMC model by considering Delaunay triangulation of the parameter space of a higher order Markov chain.

The algorithm in Dahl et al. (2009) is guaranteed to identify the posterior mode clustering in the space of partitions under the constraint that sufficient statistics are univariate for each data entity and can thus be unambiguously ordered before starting the search of the mode. In addition, the likelihood terms for each data entity must satisfy certain conditions such that posterior optimal clusters are represented by contiguous subsets in the ordered data collection. The key benefit from this is that the subsequent recursive dynamic programming approach avoids unnecessary search steps in the partition space by exclusion of cluster mergings known to lead to inferior partitions in terms of posterior probability. We introduce a similar idea in higher dimensions by first using Delaunay triangulation to create a neighborhood structure in the SMC parameter space and then optimizing the partition in terms of posterior probability by using local operations based on Bayes factors. Unlike Dahl's algorithm, this is not guaranteed to converge to the global mode of the posterior. However, we show that our method yields a consistent estimate of the SMC structure in a local neighborhood when the sequence length tends to infinity. We demonstrate that our algorithm is considerably faster than the stochastic optimization used by Jääskinen et al. (2013) and that it outperforms also k-means and nearest neighbor algorithms both in terms of speed and accuracy.

The outline of the paper is as follows. In Section 2, we review the SMC models introduced in Jääskinen et al. (2013) and describe how Bayesian inference can be performed for them. The clustering algorithm based on Delaunay triangulation is introduced in Section 3. In Section 4, the performance of the algorithm is illustrated with both synthetic data and bacterial DNA sequence data. Some remarks about possible further developments of the algorithm, in particular in the context of bioinformatics applications, are given in the final section.

2 Sparse Markov chain models

We first briefly review the basic characteristics of sparse and variable-length Markov chains using definitions and examples from Jääskinen et al. (2013). Let $\{X_t\}_{t=1}^n$ be a se-

quence of random variables X_t which take values from a finite alphabet $\mathcal{X} = \{1, \ldots, J\}$. Assume that such a sequence can be represented by a time homogeneous Markov chain of order m denoted by $MC(m)$. The set which contains all the possible values of the subsequence $(X_{t-1}, \ldots, X_{t-m})$ is denoted by $\mathcal{X}^m$. For a state $i \in \mathcal{X}^m$ we denote the transition probability distribution $P(X_t|(X_{t-1}, \ldots, X_{t-m}) = i)$ by a vector $\mathbf{p}_{i|\cdot}$. Note that in accordance with some of the literature on Markov chains, the state i is placed on the left side of the condition rather than on the right side as in the standard notation for conditional probability.

Definition 1 (Sparse Markov chain (SMC))**.** *Consider a time homogeneous Markov chain $MC(m)$ of order m. Let $S = \{s_1, \ldots, s_k\}$ be a partition of $\mathcal{X} = \{1, \ldots, J\}^m$ such that $\mathbf{p}_{i|\cdot} = \mathbf{p}_{j|\cdot} = \boldsymbol{\theta}_c$ for all pairs of $\{i, j\}$, $i, j \in s_c$, $c = 1, \ldots, k$, and $\mathcal{P} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k\}$ is the set of k distinct transition probability vectors. The pair $(S, \mathcal{P})$ forms a sparse Markov chain model from $MC(m)$.*

By definition, an SMC merges identical transition probability vectors of a Markov chain into invariance classes, which reduces the effective parametric dimension of the model. Since the number of parameters in a higher order Markov chain increases exponentially as a function of the order, it can be essential to use a sparser model class to reduce noise in the parameter estimates. Also, characteristics of the invariance classes convey important structural information about context-specific independence embedded in a Markov chain. These properties are illustrated by the following examples.

Example 1. *Let $MC(2)$ be an ordinary Markov chain of order 2, which takes values from the DNA alphabet $\mathcal{X} = \{A, C, G, T\}$, and define partition S according to*

$$S = \{\{AA, AC, AG, AT\}, \{CA, CC, CG, CT\}, \{GA, GC, GG, GT\}, \{TA, TC, TG, TT\}\}.$$

Such a partition corresponds a set of transition probability vectors $\mathcal{P} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \boldsymbol{\theta}_3, \boldsymbol{\theta}_4\}$, where

$$\begin{aligned}
\boldsymbol{\theta}_1 &= p_{AA|\cdot} = p_{AC|\cdot} = p_{AG|\cdot} = p_{AT|\cdot}, \\
\boldsymbol{\theta}_2 &= p_{CA|\cdot} = p_{CC|\cdot} = p_{CG|\cdot} = p_{CT|\cdot}, \\
\boldsymbol{\theta}_3 &= p_{GA|\cdot} = p_{GC|\cdot} = p_{GG|\cdot} = p_{GT|\cdot}, \\
\boldsymbol{\theta}_4 &= p_{TA|\cdot} = p_{TC|\cdot} = p_{TG|\cdot} = p_{TT|\cdot}.
\end{aligned}$$

The SMC $(S, \mathcal{P})$ in Example 1 implies that for each X_n the preceding state X_{n-1} is irrelevant for predicting the state of X_n, whereas X_{n-2} is always relevant, whatever the state of X_{n-2}. Ignoring the probability distribution of the initial state as is typical for estimation of Markov chain parameters, the SMC model in the example reduces the number of free parameters of the corresponding MC from 48 to 12.

Example 2. *Let $MC(2)$ be an ordinary Markov chain of order 2, which takes values from the DNA alphabet $\mathcal{X} = \{A, C, G, T\}$, and define partition S according to*

$$\begin{aligned}
S = \{&\{A\}, \{GT, TT\}, \{AC\}, \{CC\}, \{GC\}, \{TC\}, \{AG\}, \{CG\}, \{GG\}, \\
&\{TG\}, \{AT\}, \{CT\}\}
\end{aligned}$$

such that $\mathcal{P} = \{\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_{12}\}$*, where the two first vectors are*

$$\boldsymbol{\theta}_1 = p_{AA|\cdot} = p_{CA|\cdot} = p_{GA|\cdot} = p_{TA|\cdot},$$
$$\boldsymbol{\theta}_2 = p_{GT|\cdot} = p_{TT|\cdot}.$$

In Example 2, the model can also be represented by the so-called context model (Rissanen et al., 1983; Bühlmann et al., 1999), where for each X_t the finite history $(X_{t-1}, \ldots, X_{t-m})$ is mapped to the shortest possible context dependent subset $(X_{t-1}, \ldots, X_{t-r}), r \leqslant m$, such that

$$P(X_t|X_{t-1} = x_{t-1}, \ldots, X_{t-m} = x_{t-m}) = P(X_t|X_{t-1} = x_{t-1}, \ldots, X_{t-r} = x_{t-r}). \quad (1)$$

This kind of a model is generally referred to as variable length Markov chain (VLMC). A VLMC model represents a special case of an SMC model, when the partition of transition probability vectors satisfies certain conditions such that the classes can be unambiguously mapped to a context-tree. For further details about the connection between SMC and VLMC models, see Jääskinen et al. (2013). The example below illustrates an additional case of an SMC which is also a VLMC model, such that all the additional independences imposed by the model are context-specific and alternatively representable through a context-tree. For a comprehensive discussion about context-trees, see Bühlmann et al. (1999).

Example 3. *Let* $MC(3)$ *be an ordinary Markov chain of order 3, which takes values from the binary alphabet* $\mathcal{X} = \{0, 1\}$*, and define partition* S *according to*

$$S = \{\{000\}, \{100\}, \{010, 110\}, \{001, 101, 011, 111\}\}.$$

This corresponds to the following context-specific independences, which reduce the number of free parameters from 8 to 4:

$$P(X_t|X_{t-1} = 1, X_{t-2} = x_{t-2}, X_{t-3} = x_{t-3}) = P(X_t|X_{t-1} = 1),$$

$$P(X_t|X_{t-1} = 0, X_{t-2} = 1, X_{t-3} = x_{t-3}) = P(X_t|X_{t-1} = 0, X_{t-2} = 1).$$

Consider an SMC model defined by the pair $(S, \mathcal{P})$, where we have k vectors of parameters $\{\mathbf{p}_{c|\cdot} : c = 1, \ldots, k\}$. An analytical expression for the marginal likelihood of observed sequence data given S was derived by Jääskinen et al. (2013). Below we recite their result for later use in the recursive learning algorithm. Let $\theta \in \Theta$ denote collectively the set of quantitative parameters of an SMC model. A conjugate multivariate Dirichlet prior for the matrix of transition probabilities (see, e.g., Koski, 2001) has the expression

$$p(\theta|\alpha, q) = \prod_{c=1}^{k} \left[\frac{\Gamma(\alpha)}{\prod_{j=1}^{J} \Gamma(\alpha q_j)} \prod_{j=1}^{J} p_{c|j}^{\alpha q_j - 1} \right], \quad (2)$$

where the hyperparameters satisfy the following conditions: $\alpha > 0, q_j > 0, \sum_{j=1}^{J} q_j = 1$. The likelihood of an observed data sequence $\mathbf{x} = x_0 x_1 \cdots x_n$ with initial state $z_0 =$

$(x_0 x_1 \cdots x_{m-1})$ assumed fixed equals under the SMC model

$$p(\mathbf{x}|\theta, S) \propto \prod_{i \in \mathcal{X}^m} \prod_{j=1}^{J} p_{i|j}^{n_{i|j}} = \prod_{c=1}^{k} \prod_{j=1}^{J} p_{c|j}^{n_{c|j}}, \tag{3}$$

where $n_{i|j}$ is the observed count of transitions from the state i to j in $\mathbf{x}$ and $n_{c|j} = \sum_{i \in s_c} n_{i|j}$. Consequently, the marginal likelihood $p(\mathbf{x}|S)$ of $\mathbf{x}$ is available analytically using the properties of Dirichlet distribution, such that

$$\begin{aligned} p(\mathbf{x}|S) &\propto \int_{\theta \in \Theta} p(\mathbf{x}|\theta, S) p(\theta|\alpha, q) d\theta \\ &\propto \int_{\theta \in \Theta} \left[\prod_{c=1}^{k} \frac{\Gamma(\alpha)}{\prod_{j=1}^{J} \Gamma(\alpha q_j)} \prod_{j=1}^{J} p_{c|j}^{\alpha q_j - 1} \prod_{j=1}^{J} p_{c|j}^{n_{c|j}} \right] d\theta \\ &\propto \prod_{c=1}^{k} \frac{\Gamma(\alpha)}{\prod_{j=1}^{J} \Gamma(\alpha q_j)} \frac{\prod_{j=1}^{J} \Gamma(n_{c|j} + \alpha q_j)}{\Gamma((\sum_{j=1}^{J} n_{c|j}) + \alpha)}, \end{aligned} \tag{4}$$

where $\Gamma(\cdot)$ is the Gamma function. Inference about S can be done using the posterior distribution

$$p(S|\mathbf{x}) \propto p(\mathbf{x}|S) p(S). \tag{5}$$

In our numerical experiments, a uniform prior $p(S) = 1/B(|\mathcal{X}|^m)$ is for simplicity assigned over the space of all possible partitions of $\mathcal{X}^m$ to obtain the posterior probability of S, where $B(n)$ is the Bell number of n. However, alternative priors could also be deployed. For example, a Dirichlet process (DP) prior assigns the following probability on the partitions

$$p(S) \propto \beta^k \prod_{c=1}^{k} \Gamma(|s_c|), \tag{6}$$

where β is a concentration parameter governing the implied probability mass over possible values of k. A uniform prior on k, given an upper limit $K \leq |\mathcal{X}^m|$, distributes evenly the same probability mass $1/K$ over all partitions with a given number of classes k. Since the number of ways of partitioning a set of $|\mathcal{X}^m|$ elements into k non-empty subsets is given by the Stirling number of the second kind, the probability of a partition S with k clusters equals $p(S) = K^{-1} \left\{ {|\mathcal{X}^m| \atop k} \right\}^{-1}$. Both of these alternative priors imply penalties for any particular partition when k increases and therefore favor the partition with smaller k. It should nevertheless be noted that the Dirichlet prior for the transition probability vectors already imposes an increasing penalty as a function of the number of clusters in the partition.

Given the sequence $\mathbf{x} = x_0 x_1 \cdots x_n$, the predictive likelihood for a future observation $\mathbf{y} = y_1 \cdots y_m$ under an SMC model $(S, \mathcal{P})$ can be expressed as

$$\begin{aligned} &p((Y_m = y_m, \ldots, Y_1 = y_1)|(X_n = x_n, \ldots, X_0 = x_0), S) = \\ &= \int_{\theta \in \Theta} p(\mathbf{y}|\mathbf{x}, \theta, S) p(\theta|\mathbf{x}, S) d\theta \end{aligned} \tag{7}$$

$$= \prod_{c=1}^{k} \frac{\Gamma(\alpha + m_{c|j})}{\Gamma((\sum_{j=1}^{J} m_{c|j} + n_{c|j}) + \alpha)} \prod_{j=1}^{J} \frac{\Gamma(m_{c|j} + n_{c|j} + \alpha q_j)}{\Gamma(m_{c|j} + \alpha q_j)},$$

where $m_{i|j}$ is defined analogously to the count $n_{i|j}$ and is calculated from the sequence $\mathbf{y}$.

3 Learning SMC models from observed sequence data

Jääskinen et al. (2013) used stochastic optimization to estimate posterior mode partition of states for an SMC model. Their algorithm uses search operators typical to Bayesian clustering methods where the number of clusters is not fixed in advance. The local moves in the partition space consist of merge and split of existing clusters, as well as re-assignment of individual data entities among clusters. In contrast to purely stochastic proposals which are inefficient for finding reasonable splits of clusters, the algorithm uses deterministic hierarchical clustering within an existing cluster to obtain informed proposals which are more likely to yield an improvement to the log-unnormalized posterior. The re-assignment of individual data entities is attempted in a random order, but otherwise the algorithm proceeds in a greedy manner by choosing the local move that leads to maximal increase in the log-unnormalized posterior. Jääskinen et al. (2013) showed that this approach did yield good accuracy for the partition estimation.

In Dahl et al. (2009), a recursive dynamic programming algorithm was introduced, such that the posterior modal partition can be found in a deterministic manner for a set of data entities under a clustering model. A main advantage of the algorithm is to exploit convexity of clusters, which avoids unnecessary trials of cluster merging and splitting of the type discussed above. Inspired by this, we first determine neighborhoods for Markov chain transition probability vectors using Delaunay triangulation. These neighborhoods are then used in a recursive fashion to identify an approximate posterior mode partition over the space of clusterings of Markov chain states.

Delaunay triangulation is a mathematical tool used mainly for constructing a topology for a given set of data points. For a given set P of data points, a Delaunay triangulation of P is defined as a triangulation $DT(P)$ such that no point in P is inside the circum-hypersphere of any simplex in $DT(P)$. Delaunay triangulation was first discussed by Boris Delaunay and has been widely used in a variety of applications (De Berg et al., 2000). Particularly, it has gained popularity in spatial clustering problems by providing a compact proximity between the data points (Yang and Cui, 2008), such that the nearest points are always connected as neighbors in the triangulation (Liu et al., 2008; Deng et al., 2011).

For a given sequence $\{X_t\}_{t=1}^{n}$, learning an optimal SMC can be interpreted as finding a partition $S = \{s_1, \ldots, s_k\}$ on $\mathcal{X}^m$, where the posterior defined in (5) is maximized and then obtaining an estimate of the transition probability distribution for all classes in the partition S (alternatively, the posterior predictive distribution in (7) could also be used). We can consider the ordinary Markov model $MC(m)$ as a trivial SMC model with the number of classes in the partition equal to $|\mathcal{X}|^m$. For each state $i \in \mathcal{X}^m$, values of the free parameters in the corresponding transition probability vector $\mathbf{p}_{i|\cdot}$ are coordinates in a $(J-1)$-dimensional simplex. Correspondingly, all the states in $\mathcal{X}^m$ can

then be mapped into points in such a simplex. By applying Delaunay triangulation on $\mathcal{X}^m$, each transition state $i \in \mathcal{X}^m$ becomes a node in the triangulation $DT(\mathbf{X}^m)$.

We can maximize the posterior defined in (5) by recursively merging those nodes in the Delaunay triangulation where it leads to an improvement of the posterior probability. Bayes factor appears to provide an efficient local criterion for this task. For any two clusters u, v, the Bayes factor for merging them against keeping the two states as separate classes in the model is defined as

$$\begin{aligned} BF_{uv} &= \frac{P(\mathbf{x}|M(\{u,v\}))}{P(\mathbf{x}|M(\{u\},\{v\}))} \\ &= \frac{\prod_{j=1}^{J}\Gamma(\alpha q_j)}{\Gamma(\alpha)} \frac{\prod_{j=1}^{J}\Gamma(n_{u|j}+n_{v|j}+\alpha q_j)}{\Gamma((\sum_{j=1}^{J} n_{u|j}+n_{v|j})+\alpha)} \frac{\Gamma((\sum_{j=1}^{J} n_{u|j})+\alpha)}{\prod_{j=1}^{J}\Gamma(n_{u|j}+\alpha q_j)} \frac{\Gamma((\sum_{j=1}^{J} n_{v|j})+\alpha)}{\prod_{j=1}^{J}\Gamma(n_{v|j}+\alpha q_j)} \end{aligned} \tag{8}$$

where, with a slight abuse of notation, $M(\{u,v\})$ refers the SMC model that merges u, v while $M(\{u\},\{v\}))$ refers to the model where u, v remain separate, and the remaining parts of the models are identical. The form of (8) reveals that the calculations only involve the sufficient statistics of the two clusters, which enables efficient localized computation. The theorem below establishes that the local operations converge to correct decisions as the sequence length increases, when relevant neighbors are included in the comparisons.

Theorem 1. *Let a sequence $\{X_t\}_{t=1}^n$ and a sparse Markov model $(S,\mathcal{P})$ constructed from $\{X_t\}_{t=1}^n$ be given. For any two clusters u, v in S, it holds*

$$\lim_{n\to\infty} \log BF_{uv} = \begin{cases} +\infty, & \mathbf{p}_{u|\cdot} = \mathbf{p}_{v|\cdot}, \\ -\infty, & \mathbf{p}_{u|\cdot} \neq \mathbf{p}_{v|\cdot}. \end{cases} \tag{9}$$

Proof. A detailed proof is provided in the supplementary appendix (Xiong et al., 2015). □

The robustness of Bayes factor as a criterion of merging increases with the length of the data sequence. Similarly, since the neighborhood of any state is determined by a Delaunay triangulation of consistent estimates of all the transition probability distributions, states associated with identical underlying transition probability vectors will become neighbors with an increasing probability. We propose the heuristic algorithm (Algorithm 1) for searching the optimal $(S,\mathcal{P})$ for a given sequence $\{X_t\}_{t=1}^n$.

To identify the posterior optimal m, one can use the above algorithm to learn the posterior mode conditional on each $m = 0,\ldots,M$, and store the corresponding values of $p(\mathbf{x}|S)$ and $p(S)$. Let $p(\mathbf{x}|S_m)$ denote the marginal likelihood and $p(S_m)$ the prior probability for a particular value of m, under the assumption that the initial observations $x_0 x_1 \cdots x_{M-1}$ are considered fixed to ensure compatibility of model comparisons between all putative values of m. Then, an optimal value of m and the corresponding partition can be obtained by the following maximization operation:

$$\hat{S} = \underset{m\in\{0,\ldots,M\}}{\arg\max} \left\{ \underset{S_m\in\mathcal{S}_m}{\arg\max}\, p(\mathbf{x}|S_m)p(S_m) \right\}, \tag{10}$$

where $\mathcal{S}_m$ refers to the space of possible partitions for a given order m.

Algorithm 1: Deterministic recursive learning for sparse Markov models.

Data: Input sequence $\{X_t\}_{t=1}^n$
Result: Sparse Markov model $(S, \mathcal{P})$
Calculate the transition counts from $\{X_t\}_{t=1}^n$ for $MC(m)$;
Estimate each transition probability distribution $\mathbf{p}$ of $MC(m)$ using the posterior mean based on the transition counts;
Obtain Delaunay triangulation G of $\mathcal{X}^m$ by using values of free parameters in $\mathbf{p}$ as coordinates;
Calculate the log Bayes factor $\log BF_{uv} = \log P(\mathbf{x}|M(\{u,v\})) - \log P(\mathbf{x}|M(\{u\},\{v\}))$ for each edge u, v in G;
Find the edge (u^*, v^*) with max log Bayes factor value w;
Set $\mathcal{U} = u^*$,$\mathcal{V} = v^*$ and $\mathcal{W} = w$;
while $\mathcal{W} > 0$ **do**
    Merge $\mathcal{V}$ to $\mathcal{U}$ by the following steps:
    (a) Add the sufficient statistics counts of $\mathcal{V}$ to $\mathcal{U}$;
    (b) For each node r in G which has a connection with $\mathcal{V}$, if edge $(\mathcal{U}, r)$ does not exist, redirect the edge $(\mathcal{V}, r)$ to $(\mathcal{U}, r)$;
    (c) Delete $\mathcal{V}$ from G;
    Update the Bayes factors for all the edges (include the edges added by merging) connected to $\mathcal{U}$;
    Find an edge $(u^{*'}, v^{*'})$ with max log Bayes factor value w';
    Set $\mathcal{U} = u^{*'}$,$\mathcal{V} = v^{*'}$ and $\mathcal{W} = w'$;
end

The computational complexity of Delaunay triangulation construction equals $\Theta(|\mathcal{X}|^m \log(|\mathcal{X}|^m))$ and the maximum number of edges in a triangulation is $3|\mathcal{X}|^m - 6$ (Yang and Cui, 2008; Lee and Schachter, 1980). An example output of the algorithm, together with the generating partition and estimation errors, is visualized in Figure 1 in the next section.

4 Numerical results

Since SMC models have already been compared with alternative models in an earlier work by Jääskinen et al. (2013), we focus on examining the predictive performance of Delaunay triangulation based SMC estimates and on comparing the speed and accuracy with the stochastic optimization algorithm and popular standard clustering algorithms. First, we applied the deterministic recursive algorithm to data generated by a fifth order SMC model. We simulated DNA data sequence with a length of 1000000 base pairs from a fifth order SMC with 100 invariance classes for training purpose. The model learning was performed with the maximum order ranging from 3 to 7. The parameter values of the optimized SMC model were obtained by MAP estimation and then used to calculate the predictive log-loss on additional 10 sequences generated from same

generating mechanism:

$$l(\hat{P}, x) = -\frac{1}{T}\sum_{i=1}^{T} \log_2 \hat{P}(x_i|x_1, \ldots, x_{i-1}). \tag{11}$$

To generate the synthetic data under an SMC of order 5 with 100 invariance classes, parameters for the generating model were sampled as follows. First, we sampled a vector $\mathbf{z}_l = (z_{l1}, z_{l2}, z_{l3}, z_{l4})$, with $l = 1, \ldots, 100$ from a four-dimensional multivariate normal distribution with zero mean vector and covariance matrix defined as $5 \cdot I_4$ (I_4 denoting a 4×4 identity matrix). Distributions of the transition probability vectors were then sampled from the corresponding Dirichlet$(e^{z_{l1}}, e^{z_{l2}}, e^{z_{l3}}, e^{z_{l4}})$ for each of the 100 classes. These transition probability vectors were assigned to the 1024 states of the fifth order MC with uniform probabilities. In addition to the log-loss, we used the Adjusted Rand Index (ARI) (Hubert and Arabie, 1985) to measure the similarity between the estimated posterior mode partition and the generating partition. ARI is also used later in several comparisons of algorithm performance.

Order	$\lvert\mathcal{X}\rvert^m$	ARI	CPU-time (s)	log-loss
3	64	0.0779	0.0716	1.9567($\pm$0.0004)
4	256	0.1388	0.3673	1.8575($\pm$0.0008)
5	1024	0.9570	2.5184	1.5496($\pm$0.0007)
6	4096	0.6455	15.9570	1.5562($\pm$0.0007)
7	16384	0.3327	88.4484	1.5890($\pm$0.0011)

Table 1: Results from optimizing a fifth order SMC model with different maximum orders. The standard deviations (SD) of log-losses across replicates are given in parentheses.

Predictive performance of the suggested algorithm is summarized in Table 1. It is seen that the algorithm tends to find a reasonable partition structure when the maximum *a priori* eligible order is equal to or higher than the order of the generating model. However, the algorithm may still be affected by noise resulting from the redundant memory. A strategy commonly used in machine learning for noise control is to perform model order selection by out-sample predictive validation. By splitting the available data into training and test sets and calculating the log-loss of the test data using the trained model, an order can be selected by choosing the model with minimal log-loss instead of maximizing the posterior probability over the *a priori* eligible orders. The adequate performance of this procedure is illustrated in Table 1 where log-loss is minimized for the correct order.

Second, we compare the deterministic recursive algorithm with the stochastic greedy algorithm Jääskinen et al. (2013). In this experiment we investigated how an increase in the amount of training data influences the model learning. We generated DNA sequences from SMC models with orders 3 and 4. The numbers of invariance classes for the two models are 16 and 64, respectively. The different sequence lengths considered were 10000, 100000 and 1000000. For simplicity, in this experiment the order of the SMC model is assumed to be known. The results are shown in Tables 2 and 3. Generally, an increase in

the model order requires more data for training, and we can see that the deterministic recursive learning algorithm achieves comparable or occasionally even better results than the stochastic greedy algorithm in terms of accuracy. The CPU-time values in Table 3 show that the deterministic algorithm is much faster than the stochastic algorithm, even up to several orders of magnitude when higher orders and longer data sequences are analyzed.

3rd Order, 16 Populations, $\|\mathcal{X}\|^m = 64$		
Length	SMC-D	SMC-S
10000	0.923(0.037)	0.916(0.029)
100000	0.999 (0.003)	0.999(0.003)
1000000	1(0)	1(0)
4th Order, 64 Populations,$\|\mathcal{X}\|^m = 256$		
Length	SMC-D	SMC-S
10000	0.537(0.025)	0.545(0.025)
100000	0.939(0.015)	0.944(0.013)
1000000	0.998(0.003)	0.998(0.003)

Table 2: Average ARI (SD in parentheses) for simulated data sets with different order and sequence length. Each result is based on 10 simulation runs on a single core with 3.4 GHz processor. SMC-S, Sparse Markov chain using stochastic greedy optimization; SMC-D, Sparse Markov chain using deterministic recursive algorithm.

3rd Order, 16 Populations, $\|\mathcal{X}\|^m = 64$		
Length	SMC-D	SMC-S
10000	0.106(0.004)	4.690(0.340)
100000	0.107(0.003)	23.6490(2.672)
1000000	0.107(0.004)	237.191(43.449)
4th Order, 64 Populations,$\|\mathcal{X}\|^m = 256$		
Length	SMC-D	SMC-S
10000	0.537(0.007)	24.784(3.401)
100000	0.502(0.005)	74.522(8.716)
1000000	0.493(0.012)	511.398(48.696)

Table 3: Average CPU running time in seconds (SD in parentheses) for simulated data sets with different order and sequence length. Each result is based on 10 simulations.

We further compare with the popular nearest neighbor (NN) search and k-means clustering algorithms (Bishop, 2007). Since neither of these algorithms can automatically determine the number of clusters, they are modified as follows. The algorithm used for NN search is described in detail in the supplementary appendix (Xiong et al., 2015). In short, we use Bayes factor as the distance metric and apply a similar updating approach as described in Algorithm 1. As NN is fully deterministic, no initialization needs to be specified. The reason for using the Bayes factor also for NN is the need for being able to determine the number of clusters automatically. Otherwise with hierarchical clustering partitions had to be obtained by cutting from level to level.

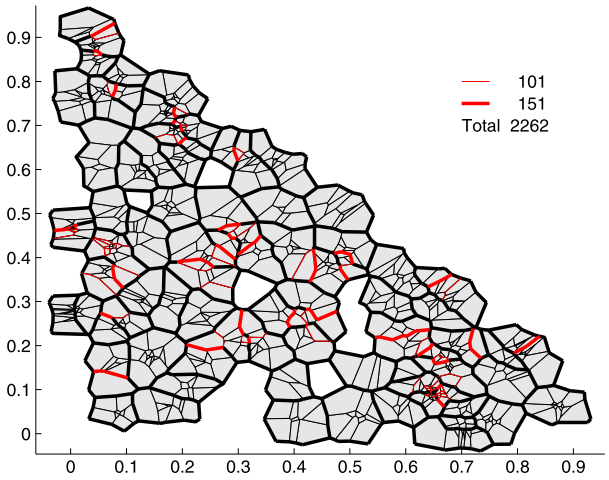

Figure 1: Partition estimated by the SMC-D algorithm. The thick black lines represent boundaries between invariance classes, and the thin black lines indicate triangulation cells. The red lines indicate errors in the estimated partition (thick line, erroneous split; thin line, erroneous merging). The number of edges in the triangulation is 2026; however, the lines on the boundary of the graph do not represent edges of the Delaunay triangulation, which brings the total number of elements considered in the error calculation to 2262.

For the k-means clustering, we use a standard implementation such that each possible value of k from the minimum to maximum is tested and the solution with the highest log-unnormalized posterior is chosen. The k-means algorithm is reasonably fast for small values of k; however, its convergence time increases very steeply for moderate to large values, which causes a prohibitively long total execution time for the higher order models considered. For illustration, we generated sequences of length 1000000 with $|\mathcal{X} = 3|$ from a sixth order Markov chain with 81 invariance classes. For each transition probability vector of the model, we estimate its value by posterior mean and project the first two coordinates of the estimate on a 2D space. All codes used in these experiments are available for download from `http://www.helsinki.fi/bsg/filer/SMCD.zip`.

Figure 1 shows how the generating partition relates to the Delaunay triangulation of the posterior mean estimates of the transition probability vectors. For a majority of invariance classes, using the local neighborhood results in finding the relevant states that should be merged together in the partition. Comparison with Figures 2 and 3 illustrates that the SMC-D algorithm outperforms the other two algorithms in terms of erroneous splits and mergings of invariance classes in the resulting mode partition estimate. The three figures show the error frequencies separately for splits and mergings, revealing that the Delaunay triangulation based approach controls best both types of errors simultaneously. However, the k-means algorithm is nearly as good regarding control of erroneous splits, whereas its error rate for merging is more than twice the rate for SMC-D. Conversely, the nearest neighbor approach has a much higher error rate for

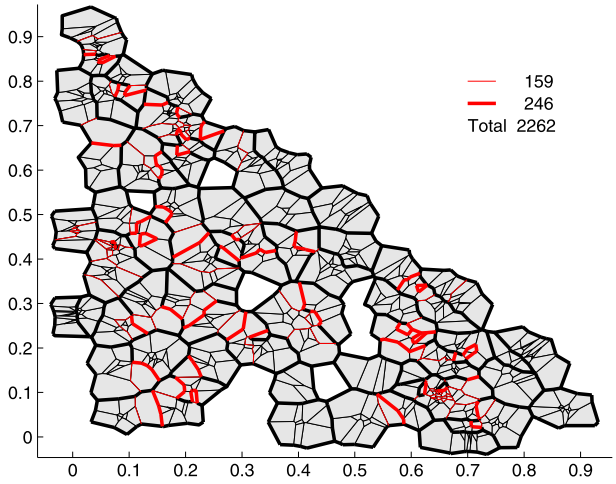

Figure 2: Partition estimated by the nearest neighbor clustering algorithm. The thick black lines represent boundaries between invariance classes, and the thin black lines indicate triangulation cells. The red lines indicate errors in the estimated partition (thick line, erroneous split; thin line, erroneous merging).

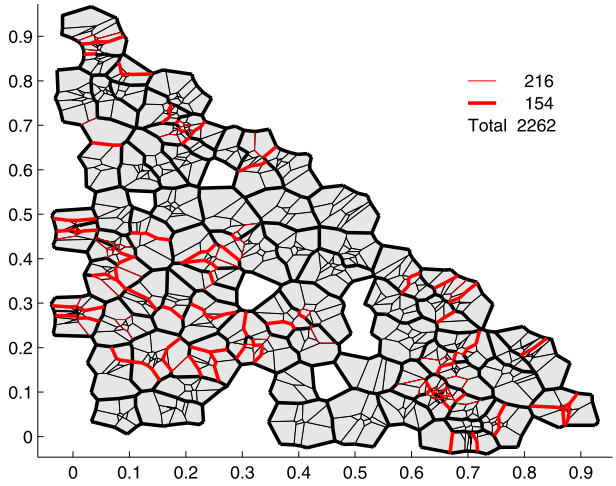

Figure 3: Partition estimated by the k-means algorithm. The thick black lines represent boundaries between invariance classes, and the thin black lines indicate triangulation cells. The red lines indicate errors in the estimated partition (thick line, erroneous split; thin line, erroneous merging).

the splits, often producing small false clusters within a single true invariance class while having a more reasonable control of the merging errors.

To compare the three deterministic algorithms in more challenging cases, we generated DNA sequences from SMC models with orders 5 and 6. The numbers of invariance classes for the models are 256 and 1024, respectively. The different sequence lengths considered were 10000, 100000 and 1000000. Tables 4 and 5 demonstrate that our algorithm outperforms the neighbor-joining and k-means algorithms both in terms of computational scalability and accuracy of the estimates. Note that k-means results were not obtained for the sixth order model due to the excessive run times of the algorithm compared with the two others.

5th Order, 256 Populations,$\lvert\mathcal{X}\rvert^m = 1024$			
Length	SMC-D	NN	K-means
10000	0.4364(0.0206)	0.2020(0.0140)	0.4076(0.0173)
100000	0.8989(0.0149)	0.7297(0.0225)	0.8002(0.0154)
1000000	0.9948(0.0037)	0.9660(0.0092)	0.7961(0.0234)
6th Order, 1024 Populations,$\lvert\mathcal{X}\rvert^m = 4096$			
Length	SMC-D	NN	K-means
10000	0.1070(0.0031)	0.0178(0.0006)	NA
100000	0.4130(0.0090)	0.1550(0.0056)	NA
1000000	0.8911(0.0080)	0.6844(0.0120)	NA

Table 4: Average ARI (SD in parentheses) for simulated data sets with varying order and sequence length. Each result is based on 10 simulations. NN, nearest neighbor search; K-means, k-means clustering algorithm.

5th Order, 256 Populations,$\lvert\mathcal{X}\rvert^m = 1024$			
Length	SMC-D	NN	K-means
10000	4.9877(0.2190)	8.0962(1.9275)	668.7770(122.0522)
100000	4.5356(0.6128)	7.6964(2.3118)	481.5835(71.2533)
1000000	4.2605(0.1179)	7.0981(1.8584)	344.9480(51.5290)
6th Order, 1024 Populations,$\lvert\mathcal{X}\rvert^m = 4096$			
Length	SMC-D	NN	K-means
10000	39.1019(5.6826)	257.1691(58.2820)	NA
100000	34.2587(2.9504)	325.2469(80.7503)	NA
1000000	29.7716(2.1591)	278.7225(67.1614)	NA

Table 5: Average CPU running time (SD in parentheses) for simulated data sets with different order and sequence length. Each result is based on 10 simulations.

Finally, to illustrate the performance of our algorithm on real DNA data, we expand the experiment reported in Table 1 of Jääskinen et al. (2013). The experiment used a large bacterial genomic database investigated in detail in Corander et al. (2012), where each sequence was assigned to a cluster using a population genomic model. The experiment was performed using the concatenated multilocus sequence typing (MLST)

DNA sequences of 7829 *Neisseria meningitidis* bacterial strains. In the example, 3915 sequences were randomly sampled from the whole dataset to train the models, and the log-loss was calculated on the remaining 3914 test sequences by using the models: Markov chain, DCTW, PPMC (Begleiter et al., 2004) and SMC. The SMC models in the original example were trained by using the stochastic greedy algorithm. Here, we report results based on applying the deterministic recursive algorithm for learning SMC on the same data. The results are shown in Table 6. The models obtained from deterministic recursive algorithm have similar performance to the models trained by stochastic greedy algorithm, which agrees with our earlier experiments on synthetic data. We also note that while the nearest neighbor method achieves here a similar level of accuracy, its log-loss is still consistently slightly higher than for SMC-D for all considered model orders.

Order	MC	PPMC	DCTW	NN	SMC-S	SMC-D
5	1.6291	1.4034	1.4243	1.3755	1.3748	1.3678
6	1.5420	0.9305	0.9285	0.8819	0.8830	0.8765
7	1.5058	0.4925	0.4834	0.4506	0.4490	0.4476
8	1.5045	0.2467	0.2283	0.2094	0.2091	0.2072
9	1.5133	0.1286	0.1195	0.1073	0.1081	0.1057
10	1.5262	0.0905	0.0809	0.0723	0.0739	0.0712

Table 6: Log-loss for predicting the 3914 concatenated multilocus sequence typing DNA sequences based on 3915 training sequences. DCTW, decomposed context tree weighting; MC, Markov chain; PPMC, prediction by partial match method-C; NN, nearest neighbor clustering; SMC-S, Sparse Markov chain using stochastic greedy optimization; SMC-D, Sparse Markov chain using deterministic recursive algorithm.

5 Discussion

The efficiency of sparse-type Markov chain models of high order has been demonstrated in applications of data compression and prediction, especially in the bioinformatics context. Earlier work derived a Bayesian representation for the SMC models which lump transition probabilities into invariance classes such that the resulting models need not have a hierarchical structure as is in general required in context tree-based approaches (Jääskinen et al., 2013). Our main finding is that Delaunay triangulation of the parameter space combined with local Bayesian model comparison offers an accurate and scalable basis for deterministic clustering that targets to approximate the posterior mode partition. Since the algorithm almost never failed to find at least an equally good partition as the stochastic optimization, it is particularly promising for applications where minimizing computation time is important. The simulation results visualized in Figures 1–3 provide some basis for understanding why the Delaunay triangulation based approach performs better than popular deterministic algorithms such as NN and k-means. The latter struggled to maintain a good control of erroneous mergings of states, which may be caused by cluster centers tending to get repelled from each other. When many clusters underlie the data, it is more challenging for k-means to balance both the local and global optimality of clusters as illustrated, for instance, by Marttinen et al. (2009).

NN performed much better than k-means in terms of erroneous mergings, but resulted instead in substantially larger number of erroneous splits. This is plausibly caused by the very definition of the algorithm, as it only acts extremely locally and does not consider among a reasonable set of alternative steps to be taken. Our recursive algorithm benefits from gaining knowledge of the most likely relevant neighbors among which clusters may be formed. Since in a typical setting majority of the states would not belong to the same class as most other states, the Delaunay triangulation based neighborhood structure effectively prevents the algorithm from trying out a large number of search operations that are likely to fail in terms of posterior support. For successful local operations, it appears essential that the relevant sufficient statistics are updated after each change to a cluster, in contrast to basic distance based methods which operate only with pre-calculated distances to determine splits and mergings.

In our approach, we used a default uniform prior for the partitions of sequence states combined with an uniform prior for the MC order. The experiments indicate that this choice leads to relatively satisfactory results. However, in the experiments we sometimes also observed that the posterior optimization may choose an SMC model with a higher order than it should be. The main reason is that the model complexity does not grow exponentially in SMC models and the penalty due to the prior may not compensate the reward of over-fitting the noise in the data. However, as briefly demonstrated, cross-validation with a predictive loss function could alternatively be used to optimize the model order to maintain sufficient level of noise control. On the other hand, the more advanced partition priors imposing a penalty on an increase in the number of clusters could be employed to solve the over-fitting issue, as discussed in Section 2.

DNA sequence data represents an attractive area of application for the SMC models combined with the deterministic learning method. Given its relatively low computational complexity, our algorithm could be modified to optimize an expressive class of bi-clustering models for clustering simultaneously both Markov chain states and large numbers of short DNA sequences met typically in 16S sequencing applications in microbial ecology. For instance, since the number of sequences covering the whole or nearly whole 16S gene currently available in databases is already extremely large and still rapidly increasing, fast Bayesian clustering methods would offer potential for a more accurate discovery of underlying data structures. For such big data sets only model-free clustering approaches are generally used due to the computational constraints related to Bayesian methods. Finally, as the DNA alphabet only requires the triangulation to be done in a 3-dimensional space, it would be interesting to generalize the algorithm into higher dimensional alphabets. This could be accomplished, for instance, by using either a high-dimensional Delaunay triangulation or other heuristic graph constructing algorithm for mapping the state space.

Supplementary Material

Appendix of article by Xiong, Jääskinen, and Corander (DOI: 10.1214/15-BA949SUPP; .pdf).

References

Begleiter, R., El-Yaniv, R., and Yona, G. (2004). "On prediction using variable order Markov models." *Journal of Artificial Intelligence Research*, 22: 385–421. MR2129473. 260

Bishop, C. M. (2007). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 1st edition. MR2247587. doi: http://dx.doi.org/10.1007/978-0-387-45528-0. 256

Bühlmann, P., Wyner, A. J., et al. (1999). "Variable length Markov chains." *The Annals of Statistics*, 27(2): 480–513. MR1714720. doi: http://dx.doi.org/10.1214/aos/1018031204. 247, 250

Corander, J., Connor, T. R., O'Dwyer, C. A., Kroll, J. S., and Hanage, W. P. (2012). "Population structure in the Neisseria, and the biological significance of fuzzy species." *Journal of The Royal Society Interface*, 9(71): 1208–1215. 259

Dahl, D. B., et al. (2009). "Modal clustering in a class of product partition models." *Bayesian Analysis*, 4(2): 243–264. MR2507363. doi: http://dx.doi.org/10.1214/09-BA409. 248, 252

De Berg, M., Van Kreveld, M., Overmars, M., and Schwarzkopf, O. C. (2000). *Computational geometry*. Springer. MR1763734. doi: http://dx.doi.org/10.1007/978-3-662-04245-8. 252

Deng, M., Liu, Q., Cheng, T., and Shi, Y. (2011). "An adaptive spatial clustering algorithm based on Delaunay triangulation." *Computers, Environment and Urban Systems*, 35(4): 320–332. 252

Farcomeni, A. (2011). "Hidden Markov partition models." *Statistics & Probability Letters*, 81(12): 1766–1770. MR2845887. doi: http://dx.doi.org/10.1016/j.spl.2011.07.012. 248

Gonzalez-Lopez, J. E., et al. (2010). "Minimal Markov Models." *arXiv:1002.0729*. 248

Hubert, L. and Arabie, P. (1985). "Comparing partitions." *Journal of Classification*, 2(1): 193–218. 255

Jääskinen, V., Xiong, J., Corander, J., and Koski, T. (2013). "Sparse Markov Chains for Sequence Data." *Scandinavian Journal of Statistics*. 247, 248, 250, 252, 254, 255, 259, 260

Koski, T. (2001). *Hidden Markov models for bioinformatics*, volume 2. Springer. MR1888250. doi: http://dx.doi.org/10.1007/978-94-010-0612-5. 250

Lee, D.-T. and Schachter, B. J. (1980). "Two algorithms for constructing a Delaunay triangulation." *International Journal of Computer & Information Sciences*, 9(3): 219–242. MR0585950. doi: http://dx.doi.org/10.1007/BF00977785. 254

Liu, D., Nosovskiy, G. V., and Sourina, O. (2008). "Effective clustering and boundary detection algorithm based on Delaunay triangulation." *Pattern Recognition Letters*, 29(9): 1261–1273. 252

Marttinen, P., Myllykangas, S., and Corander, J. (2009). "Bayesian clustering and feature selection for cancer tissue samples." *BMC Bioinformatics*, 10(1): 90. 260

Rissanen, J., et al. (1983). "A universal data compression system." *IEEE Transactions on Information Theory*, 29(5): 656–664. MR0730903. doi: http://dx.doi.org/10.1109/TIT.1983.1056741. 247, 250

Xiong, J., Jääskinen, V., and Corander, J. (2015). "Appendix of article by Xiong, Jääskinen, and Corander." *Bayesian Analysis*. doi: http://dx.doi.org/10.1214/15-BA949SUPP. 253, 256

Yang, X. and Cui, W. (2008). "A novel spatial clustering algorithm based on Delaunay triangulation." In: *International Conference on Earth Observation Data Processing and Analysis*, 728530. International Society for Optics and Photonics. 252, 254

Acknowledgments

Work of JC, JX and VJ was funded by the ERC grant no. 239784 and by the Academy of Finland grant no. 251170; work of VJ and JX was also funded by the FICS and FDPSS graduate schools.