FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

FEUP

Online topology free Gaussian HMM parameter estimation based on clustering

André Simões Fernandes

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Ricardo Morla (PhD)

Second Supervisor: Simon Malinowski (PhD)

July 19, 2012

Online topology free Gaussian HMM parameter estimation based on clustering

André Simões Fernandes

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Rui Maranhäo de Abreu (PhD)
External Examiner: Cecilia Maria Vasconcelos Costa Castro Azevedo (PhD)
Supervisor: Ricardo Morla (PhD)

July 19, 2012

Abstract

This dissertation first compiles a literature review and synopsis about the estimation of the parameters of Hidden Markov Models with an emphasis on incremental estimation and topology adaptation.

The current paradigm for estimation of the parameters of Hidden Markov Models usually maintains the number of states of the model constant throughout the process. Among the techniques that allow topology adaptation (i.e. that adjust the number of states during the process) there are some that are not suitable for online estimation. The others that are only allow the adaptation to occur in one direction by only increasing or only decreasing the number of states.

A novel approach for incremental estimation of the parameters of Hidden Markov Models is presented that allows continuous topology adaptation. The number of states of the model can increase, decrease or maintain depending on the incoming observations. This algorithm is well adapted for the detection of concept changes.

Acknowledgements

I want to express my deepest gratitude to my supervisors, Dr. Simon Malinowski and Dr. Ricardo Morla, for their endless patience, guidance and support which in turn allowed this dissertation to come true.

I am also thankful for the financial support provided by FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within the Framework of the Carnegie Mellon University - Portugal Program and the project "NeTS: Next Generation Network Operations and Management" CMU-PT/RNQ/0029/2009.

André Fernandes

Contents

List of Figures

LIST OF FIGURES

List of Tables

LIST OF TABLES

Abbreviations

AIC	Akaike's Information Criterion
BIC	Bayesian Information Criterion
BW	Baum-Welch Algorithm
EM	Expectation-Maximization
GEM	Generalized Expectation-Maximization
HMM	Hidden Markov Model
IBW	Incremental Baum-Welch
IBW+	Incremental Baum-Welch (improved β estimation)
LLH	Log-Likelihood
MAP	Maximum a Posteriori
MCMC	Markov Chain Monte Carlo
ML	Maximum Likelihood
MLE	Maximum Likelihood Estimation
PF	Particle Filters
PMCMC	Particle Markov Chain Monte Carlo
QHMM	Quantitatized Hidden Markov Model
SMC	Sequential Monte-Carlo

Chapter 1

Introduction

1.1 Context

This dissertation was written as part of the curriculum of the Master in Informatics and Computing Engineering at the Faculty of Engineering, University of Porto.

This dissertation was also integrated in a project whose goal was to construct a framework to detect anomalies in computer network data at the Institute for Systems and Computer Engineering of Porto.

1.2 Motivation

The motivation behind this work lies in detecting anomalies in computer network data. Today's world is interconnected like never before and computer networks provide the very fabric for the development of our *Networked Society* [Wel02].

The development of automatic tools for anomaly detection in computer networks in real time has the potential to improve the quality of the experience of its users, as well as help operators anticipate possible problems.

The project in which this work fits in, not only has the goal to create this kind of tools, but has the objective to create them in a sufficiently generic way so they can be applied in many different kinds of network technologies, including GSM (Global System for Mobile Communications), BGP (Border Gateway Protocol), DSL (Digital Subscriber Line) networks and many others.

For example, DSL operators have useful information about the synchronization loss for each customer. These sequences of data are a byproduct of many occurrences such as: the existence of noise, electromagnetic interference, cross-talk between customers and hardware/software failure on either side.

Different kinds of anomalies have different patterns of synchronization times. The rationale behind the detection is that this logic can be inverted and it is possible to infer the type of anomaly based on the patterns present in the data. In this way, detecting anomalies is equivalent to identifying patterns in the data sequences.

One possible framework that allows the desired abstraction of the underlying technology consists in a two-step approach described in figure 1.1.

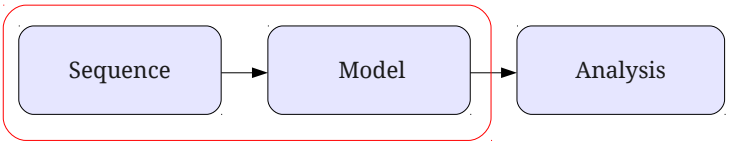

Figure 1.1: Anomaly detection steps.

Given a data set of sequences, that may include normal and abnormal behavior for instance, it might not be easy to compare these sequences directly. The modeling step aims at transforming this raw data into a format that allows the comparison of sequences to be made easier. Different models can focus on different particularities of the data. The role of the analysis step is to produce conclusions about the modeled data. Depending on the targeted application, different kind of analysis can be done. For example, clustering can be used to find similar classes in the data and outlier detection can be used to detect rare events.

Notice that the modeling and the analysis steps are completely decoupled and can be studied separately. The focus of the dissertation is on the modeling step of the sequences of data.

1.3 Goal

One approach to model time-series is through Hidden Markov Models (HMMs). HMM are able to model the time dependency in such series through a doubly embedded stochastic process.

One of the goals of this dissertation is to provide a study about ways of estimating the parameters of Hidden Markov Models by analyzing the possible alternatives and identifying possible improvements.

However, the main goal is to implement an algorithm capable of updating the parameters of Hidden Markov Models with each new observation and at the same time capable of adjusting the number of states of the model to best describe the sequences of data. These so called online free topology algorithms are desirable because batch methods become cumbersome when the number of observations increases and fixed topology algorithms are not able to adapt to changes in context of the observations.

1.4 Contributions

The main contributions of this dissertation are:

- A literature review about the algorithms that allow the estimation of the parameters of Hidden Markov Models (chapter 2).

- An adaptation of Quantization-based Hidden Markov Models for online operation and topology reconfiguration (chapter 3).
- A novel approach for estimating the parameters of Hidden Markov Models capable of topology adaptation enabling the number of states of the model to increase, decrease or maintain depending on the incoming observations (chapter 4).

These contributions were submitted in a small paper to the *1st PhD. Students Conference in Electrical and Computer Engineering* in 2012 [FMM12] and won the MSc Scientific Excellence paper award of the conference.

1.5 Organization

Chapter 2 provides a literature review about Hidden Markov Models and their parameter estimation. This chapter constitutes the related work of the domain of knowledge this work fits in. A small introduction to Hidden Markov Models is given in section 2.1. The literature review of the parameter estimation algorithms is made in section 2.2 where subsection 2.2.1 refers to offline estimation, subsection 2.2.2 refers to online estimation and subsection 2.2.3 refers to estimation with topology adaptation.

Chapter 3 discusses the Quantization-based Hidden Markov Model (QHMM) framework for offline estimation of the parameters and proposes several improvements on top of this approach in section 3.2. The improvements include a better observation space partition (subsection 3.2.1), online operation (subsection 3.2.2) and topology reconfiguration (subsection 3.2.3).

Chapter 4 proposes a novel approach for online free topology estimation of the parameters that is named ClusTrelHMM. This method extends an online clustering algorithm named ClusTrel (section 4.1) with the proposed improvements of the QHMM framework. The results of ClusTrelHMM are thoroughly compared to other algorithms and are presented in section 4.3.

Chapter 5 ends with a brief conclusion.

Chapter 2

Literature Review

2.1 Hidden Markov Models

Hidden Markov Models are widely used to describe complex probability distributions in time series because they are well adapted to capture the time dependencies in such series. This framework turned successful for modeling and classifying dynamic behaviors because it offers dynamic time warping, training algorithms and clear Bayesian semantics [BOP97].

2.1.1 Introduction

Hidden Markov Models [Rab89] (HMMs) are doubly embedded stochastic processes and are the simplest dynamic Bayesian networks capable of modeling complex probability distributions. Bayesian networks are probabilistic graphical models that represent random variables as nodes and conditional dependencies as edges in directed acyclic graphs (DAGs).

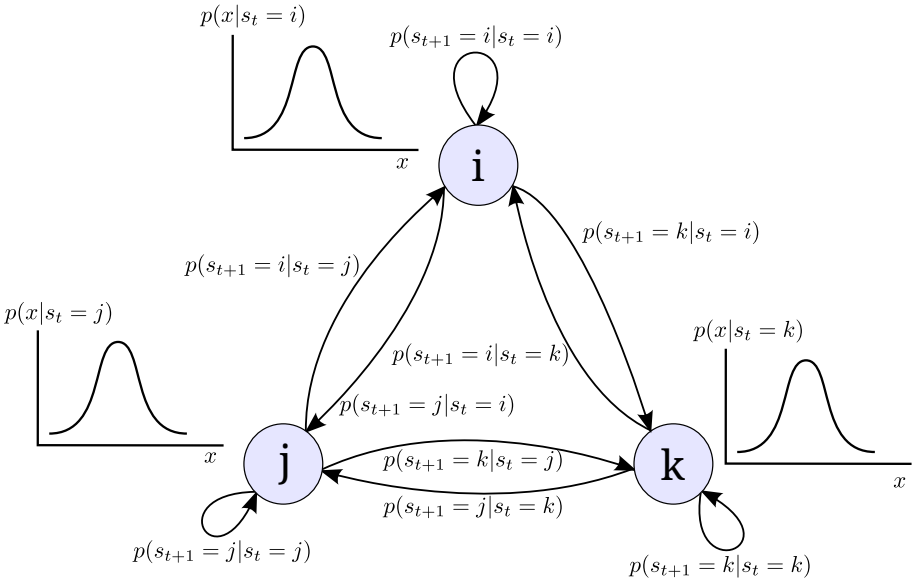

Figure 2.1: Three state Hidden Markov Model with a Gaussian distribution per state.

Each observation generated from an Hidden Markov Model depends on both stochastic processes of the model.

The first process is responsible for the selection of the state in which the observation is going to be made. This process cannot be observed directly and is usually called *hidden process* for that reason. It serves the purpose of modeling some underlying mechanism that controls the observations even if that mechanism is not well known or understood. This probabilistic distribution that is concerned with the transitions between the states is characterized by its *Markov property*, which guarantees that the transitions to future states depend only on the current state, rendering these models memoryless.

The second process controls how the observations are made in each of those states. This process can be measured directly and is therefore called the *observable process*. It requires the definition of some stochastic distribution for each of the states. These distributions are called emission distributions and can be either discrete, continuous, mixture models, etc.

Figure 2.1 shows a 3-state Gaussian HMM with states named i, j and k. Assuming we start at moment t in state i, there is a certain probability of staying in state i and two other probabilities of transitioning to state j or to state k in the next moment $t+1$. After transitioning to any one of those states an observation can be made governed by the normal distribution of that particular state as shown.

The focus of this work will be on HMMs that follow an ergodic model where any two states are connected and in models that have Gaussian or normal distribution as their emission distributions like the HMM presented in figure 2.1.

For further study please refer to [Rab89] for an introductory tutorial to discrete Hidden Markov Models with an application to speech recognition or to [Bis06] for a more in depth treatment.

2.1.2 Definition

The mathematical definition of an Hidden Markov Model requires the definition of its set of parameters. In the case of Hidden Markov Models with Gaussian distributions, the set of parameters of the model λ is a function of the transition matrix A, the initial state probabilities π, the means μ and the standard deviations σ of the normal distributions.

$$\lambda = \{A, \mu, \sigma, \pi\} \tag{2.1}$$

For an HMM with N number of states, the transition matrix $A = \{a_{ij}\}$ is a matrix of size $N \times N$ whose elements a_{ij} represent the probabilities of transitioning from state i at time t to state j at time $t+1$.

$$a_{ij} = p(s_{t+1} = j | s_t = i) \tag{2.2}$$

The means $\mu = \{\mu_i\}$ and standard deviation $\sigma = \{\sigma_i\}$ are vectors of size N that completely define the emission distributions $B = \{b_i\}$ of each state.

$$b_i = \mathcal{N}(\mu_i, \sigma_i) \tag{2.3}$$

Lastly, the initial probabilities $\pi = \{\pi_i\}$ can be condensed in a vector of size N whose elements π_i denote the probability of initializing the model at state i.

$$\pi_i = p(s_1 = i) \tag{2.4}$$

2.1.3 Operations

The definition of the set parameters λ of the model can be used to generate sequences that follow the established distributions, but there are three other operations that make HMMs useful in real-world applications.

Given a model and a sequence it is possible to compute the probability or likelihood that this sequence has been generated by the model. This is achieved by the *Forward Algorithm* [Rab89] [Bis06, chap. 13] which basically computes, by dynamic programming, the probability of every possible path of states to generate the sequence. The main idea behind Forward Algorithm lays on the definition of the Forward Variable (also known as β variable) described in section 2.2.1.1 where the Baum-Welch algorithm is described.

The *Viterbi Algorithm* [Rab89] [Bis06, chap. 13] finds the most probable sequence of hidden states, by dynamic programming, given an observation sequence and a model.

And most importantly, we can adjust the parameters of HMMs to maximize the probability of the model to generate any given sequence numbers. The estimation of the parameters of an HMM that best describe a time series is one of the main problems related to HMM. These algorithms are the subject of the study and, as such, are the topic of section 2.2.

Another operation that is not related to an operation that can be made with a model, but nevertheless interesting because it makes the clustering of sequences using HMMs feasible is the computation of distances between two different sequences modeled by two different HMMs. In [Rab89] it's shown how a simple asymmetric distance metric can easily be turned into a symmetric one. And [PBM02] shows how more complicated metrics can be applied like *Kullback-Leibler information number* and the *BP metric*.

2.2 HMM Parameter Estimation

In this section, the focus will be on the estimation of the parameters of the HMMs to best describe a time-series.

Given a sequence of observations $O = O_1, O_2, ..., O_T$ of length T the parameters of an HMM can be adjusted to maximize the probability that the sequence is generated by the model. In

the case of Gaussian HMMs with N states, the parameter estimation determines N means and N standard deviations of the Gaussian distributions and $N(N-1)$ transition probabilities.

In the literature there are numerous ways to estimate the parameters of an HMM given a series of observations (see tables A.2 and A.2). The estimation can be classified by the approach to the problem as *Maximum Likelihood Estimation* or *Bayesian Estimation*. The estimation can also be categorized by the way the observations are processed, either in a *batch* or *incrementally*. In terms of the evolution of the topology of the model the estimation can either maintain a *fixed topology* or allow a *free topology*.

The Maximum Likelihood Estimation tries to find the parameters of the model that maximize the probability of the observed data. This is achieved by maximizing a likelihood function. In the Bayesian Estimation, the parameters are considered random and assigned a suitable prior distribution. The goal is to characterize a posterior distribution given the observations. In both of these types of estimations there are batch and incremental methods.

Batch or offline methods require all the sequence of the observations to be available before the estimation procedure begins. Incremental or online methods are able to process each observation at a time allowing the model to be updated iteratively. Incremental methods are suitable when there are real time constraints that prevent the whole sequence to be available.

Most of the methods available for HMM parameter estimation maintain the number of states of the model fixed throughout the process. Nevertheless, there are methods that adjust the topology of the model by increasing or decreasing the number of states.

Incremental free topology algorithms are particularly desirable because batch algorithms become cumbersome when the number of observations increases and topology free algorithms are able to adapt to changes in context of the observations.

For an overview of the characteristics of some of the most important offline and online algorithms present in the literature please refer to the tables A.2 and A.2 of the appendix A where the estimation approach of each algorithm, its strengths, its weakness and its time complexity are detailed.

2.2.1 Offline Learning

This section focuses on the Baum-Welch algorithm and Quantization-based HMMs.

The Baum-Welch algorithm is widely used and usually produces very good estimates of the parameters. This algorithm and its incremental version (subsection 2.2.2.1) served as comparisons for the novel algorithm proposed in chapter 4.

The Quantization-based HMM allowed several improvements discussed in chapter 3 which served the basis of the computation of the transition matrices in the novel algorithm presented in chapter 4.

2.2.1.1 Baum-Welch Algorithm

The Expectation-Maximization (EM) is a variational inference framework that serves the purpose of finding Maximum Likelihood (ML) solutions for models with hidden or latent variables like Gaussian Mixtures or Hidden Markov Models. The derivation of this framework can be consulted in [Bis06, chap. 9.4] or in [Bil98].

The EM framework is composed of a two-step iterative process that is mathematically proven to converge monotonically to a local maximum. In the first step, named Expectation, the expectations of the hidden states given the model are calculated. In the second step, named Maximization, the model is updated with the expectations for the hidden states produced in the first step.

The EM algorithm has computational effort and memory requirements of $O(N^2T)$ for each pass [FM98].

Baum-Welch algorithm [BE67] [BPSW70], named after its authors, was proved to be functionally equivalent to the application of the Expectation-Maximization framework to the particular case of Hidden Markov Models. This algorithm is the method of choice for parameter inference in HMMs due to its robustness and ease of implementation [Cap11].

The main idea behind Baum-Welch algorithm is that the probabilities of being in each state and the probabilities of transitioning between two states at any given instance can be estimated based on the current parameters of the model and the sequence of observations. These estimations can then be used to estimate a new set of parameters of the model that is better adjusted to the sequence of observations.

The computation of the probabilities of being in each state is based on two other probabilities that can be calculated by dynamic programming named α and β probabilities. The probability $\alpha_t(i)$ is the probability of being in state i at time t having made all the observations $O_1,..,O_t$. The $\beta_t(i)$ is the probability of being in state i at time t with the guarantee that all future observations $O_{t+1},..,O_T$ are going to be made.

$$\alpha_t(i) = P(O_1 O_2 ... O_t, s_t = i|\lambda) \tag{2.5}$$

$$\beta_t(i) = P(O_{t+1} O_{t+2} ... O_T|s_t = i, \lambda) \tag{2.6}$$

The α probabilities are computed iteratively from observation O_1 to O_T and follow the fact that any observation O_t can be made in any state of the model (equations 2.7 and 2.8).

$$\alpha_1(j) = \pi_j b_j(O_1) \qquad 1 \leq j \leq N \tag{2.7}$$

$$\alpha_t(j) = \left[\sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij}\right] b_j(O_t) \qquad 1 < t \leq T, \quad 1 \leq j \leq N \tag{2.8}$$

Similarly the β probabilities are computed iteratively, but from the last observation O_T to O_1

(figures 2.9 and 2.10).

$$\beta_T(i) = 1 \qquad 1 \leq i \leq N \tag{2.9}$$

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \qquad 1 \leq t < T, \quad 1 \leq i \leq N \tag{2.10}$$

After the computation of the probabilities α and β for each state and each moment in time we are able to compute the probability $\gamma_t(i)$ of being in state i at time t (equation 2.11). This computation corresponds to the expectation step of EM.

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{2.11}$$

Additionaly, the probability $\xi_t(i,j)$ of transitioning from state i at time t to state j at time $t+1$ can also be computed based on the same variables (equation 2.12).

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)} \tag{2.12}$$

And finally, with the probabilities γ and ξ, the new estimates for the parameters of the model can be computed through equations 2.13, 2.14, 2.15 and 2.16 that correspond to the maximization step of EM.

$$\pi_i = \gamma_1(i) \tag{2.13}$$

$$a_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{2.14}$$

$$\mu_i = \frac{\sum_{t=1}^{T} \gamma_t(i) O_t}{\sum_{t=1}^{T} \gamma_t(i)} \tag{2.15}$$

$$\sigma_i = \sqrt{\frac{\sum_{t=1}^{T} \gamma_t(i)(O_t - \mu_i)^2}{\sum_{t=1}^{T} \gamma_t(i)}} \tag{2.16}$$

The process of calculating probabilities α and β, followed by the probabilities γ and ξ and consequent model update is repeated until convergence is met.

In terms of the implementation it is important to notice that the probabilities α and β tend to zero when t increases. The resulting underflows of α can be avoided by normalizing α as shown in equation 2.17 in every step of the computation of the α variables (equations 2.7 and 2.8). Similarly, the underflow of β is avoided through normalization (equation 2.18) in every step of the computation of the β variables (equations 2.9 and 2.10).

$$\hat{\alpha}_t(i) = \frac{\alpha_t(i)}{\sum_{i=1}^{N} \alpha_t(i)} \tag{2.17}$$

$$\hat{\beta}_t(i) = \frac{\beta_t(i)}{\sum_{i=1}^{N} \beta_t(i)} \tag{2.18}$$

Notice that the changes introduced by the normalizations of α and β do not force the recomputation of γ and ξ because normalization factors are canceled in those equations (2.11 and 2.12).

2.2.1.2 Quantization-based HMM

The Baum-Welch algorithm is an offline algorithm that requires several passages through the data. One very simple approach that also estimates the parameters of HMMs but with only one passage through the data is proposed by Hervieu et al. in [HBC07] [HBC08] where HMMs are used to describe and model trajectories in video images using their curvature values at each point.

This proposed framework was named *Quantization-based HMM* (QHMM) because it looks at the complete sequence of observations and divides the space of the observations in a number of *bins*. Each bin is nothing more than an interval that corresponds to a state with an emission distribution defined *a priori*. These predefined emission distributions can then be used to compute the probabilities of each observation, which in turn enables the estimation of the transition matrix and the initial probabilities of the model.

More specifically, the proposed framework starts by fitting the observations to a Gaussian distribution like in figure 2.2. A range is defined by the 95% confidence interval of the Gaussian curve and is divided in equal parts by *equal-width binning* (in the figure: Bin_1, Bin_2 and Bin_3). Additionaly, two more bins of the same length are added before and after the ones that existed (Bin_x and Bin_y) with the goal of representing the extremes in the observations.

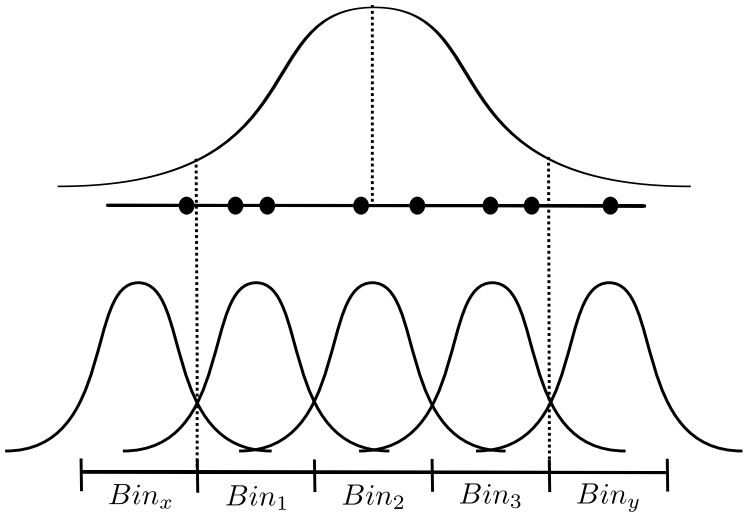

Figure 2.2: Partition of the observation space in QHMMs.

The Gaussian distributions of the states of model are defined based on the bin it corresponds. The mean of the distribution equals the center of the bin while the standard deviation equals to half of the length of the bin.

After that, the transition matrix and initial probabilities of the model can be calculated by averaging the historic of the observation probabilities and the transition probabilities as shown by equations 2.19 and 2.20:

$$a_{ij} = \frac{\sum_{t=1}^{T-1} P(O_t|s_t = i)P(O_{t+1}|s_{t+1} = j)}{\sum_{t=1}^{T-1} P(O_t|s_t = i)} \tag{2.19}$$

$$\pi_i = \frac{\sum_{t=1}^{T} P(O_t|s_t = i)}{T} \tag{2.20}$$

where $P(O_t|s_t = i)$ is the normalized probability of observing O at time t is generated by the Gaussian distribution of state i of the HMM and T is the number of observations.

The QHMM procedure presented here is based on the *Extended Least Squares* method (ELS) presented in [FM98], where the computation of the transition matrices of HMMs are seen as counting processes. The proposed algorithm for ELS in [FM98], despite the lack of convergence results, has computational complexity of only $O(N^2)$.

Please refer to section 3.1 for a comparison between BW algorithm and QHMM.

2.2.1.3 Other Offline Methods

Maximum-Likelihood Estimation Segmental K-means [JR90] is an offline method to estimate the parameters of HMMs. Instead of using the Maximum Likelihood criteria used in the EM training, it maximizes the joint likelihood of the observations and associated hidden states. The algorithm performs this optimization iteratively by alternating between finding the most likely state sequence given the current estimates of the parameters via Viterbi algorithm and obtains new estimates for the parameters by optimizing the joint likelihood of the observations and the most likely state sequence from the previous step [Dig99].

Apart from the Expectation-Maximization [BDM06] [ADST04] [OCDM06] [WSN08] that was already introduced, there are other offline methods based on MLE: the Smooth Likelihood Evaluators [Pit02], the Gradient based methods [PARM09] [PaAD06] [PDS05] and the Iterated Filtering [IaAK09].

Bayesian Estimation Some offline Bayesian algorithms are based on the Maximum a Posteriori (MAP) estimation [GHS98]. But there are other Bayesian methods based on Monte Carlo random sampling like the Particle Markov Chain Monte Carlo (Particle MCMC) [ADH09].

2.2.2 Online Learning

The previous offline algorithms have one big disadvantage, they require the complete data sequence to be available before starting the estimation of the parameters, making these algorithms useless when real-time constraints are imposed. The situation can even be worse in the case of

Baum-Welch algorithm that requires a number of undefined passages through the data until convergence occurs.

Online parameter estimation of HMMs in which the available observations are only scanned once and never stored allows continuous adaptation of the parameters along a potentially infinite data stream [Cap11].

Experimental results show that the training of incremental algorithms can be substantially faster than conventional (batch) methods and suffer no loss of recognition performance [GHS98].

2.2.2.1 Incremental Baum-Welch

To overcome the batch limitation and capitalize on the convergence properties of the BW procedure some Incremental Baum-Welch (IBW) algorithms exist in the literature [GHS98] [SRP+01] [MD08].

The main difference between BW and IBW algorithms is that the incremental version does not have access to the complete sequence of observations. As a consequence, IBW is unable to compute the β probabilities the way BW algorithm can (equations 2.9 and 2.10).

The easiest simplification capable of solving the previous problem is assuming that all β probabilities are equal to one (equation 2.21).

$$\beta_t(i) = 1 \qquad 1 \leq t \leq T, \quad 1 \leq i \leq N \tag{2.21}$$

In this case, the equations for the probabilities γ and ξ (equations 2.11 and 2.12) are simplified to the form of equations 2.22 and 2.23.

$$\gamma_t(i) = \frac{\alpha_t(i)}{\sum_{i=1}^{N} \alpha_t(i)} \tag{2.22}$$

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij}}{\sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij}} \tag{2.23}$$

The transition matrix, the means and the standard deviations of the normal distributions are estimated based on equations 2.13, 2.14 and 2.16 of the BW algorithm. These equations are algebraically manipulated to allow iterative computation as shown by the expressions 2.24, 2.25 and 2.27.

$$a_{ij}^{T} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} = \frac{\sum_{t=1}^{T-2} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} + \frac{\xi_{T-1}(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} = \frac{\sum_{t=1}^{T-2} \gamma_t(i)}{\sum_{t=1}^{T-1} \gamma_t(i)} a_{ij}^{T-1} + \frac{\xi_{T-1}(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{2.24}$$

$$\mu_i^{T} = \frac{\sum_{t=1}^{T} \gamma_t(i) O_t}{\sum_{t=1}^{T} \gamma_t(i)} = \frac{\sum_{t=1}^{T-1} \gamma_t(i) O_t}{\sum_{t=1}^{T} \gamma_t(i)} + \frac{\gamma_T(i) O_T}{\sum_{t=1}^{T} \gamma_t(i)} = \frac{\sum_{t=1}^{T-2} \gamma_t(i)}{\sum_{t=1}^{T-1} \gamma_t(i)} \mu_i^{T-1} + \frac{\gamma_T(i) O_T}{\sum_{t=1}^{T} \gamma_t(i)} \tag{2.25}$$

$$\sigma_i^T = \sqrt{\frac{\sum_{t=1}^{T} \gamma_t(i)(O_t - \mu_i)^2}{\sum_{t=1}^{T} \gamma_t(i)}} = \sqrt{\frac{\sum_{t=1}^{T-1} \gamma_t(i)(O_t - \mu_i)^2}{\sum_{t=1}^{T} \gamma_t(i)} + \frac{\gamma_T(i)(O_T - \mu_i^T)^2}{\sum_{t=1}^{T} \gamma_t(i)}} \tag{2.26}$$

$$= \sqrt{\frac{\sum_{t=1}^{T-2} \gamma_t(i)}{\sum_{t=1}^{T-1} \gamma_t(i)} (\sigma_i^{T-1})^2 + \frac{\gamma_T(i)(O_T - \mu_i^T)^2}{\sum_{t=1}^{T} \gamma_t(i)}} \tag{2.27}$$

2.2.2.2 Other Online Methods

Maximum-Likelihood Estimation Inside ML estimation there are methods that are incremental adaptations of the Segmental K-means algorithm [Dig99].

More related to EM algorithm, there are methods based on finite-memory approximations of the required smoothing computations [KM93], finite-memory approximations of the data log-likelihood [Ryd97] and pseudo-EM online algorithms [ADT05]. A recursion was also discovered in [MD08] to compute recursively the smoothing functionals required by the EM algorithm. This method was generalized in [Cap11].

An alternative to EM estimation are the online Gradient-based methods [GM97] [KDSM09] [PDS05].

Bayesian Estimation Inside Bayesian estimation there are incremental methods based on the Maximum a Posteriori (MAP) [GHS98].

Another important class of Bayesian estimators are the Sequential Monte Carlo (SMC) [KDSM09], also known as Particle Filters (PF), which are the online analogue of the Markov Chain Monte Carlo (MCMC) [ADH09].

Other methods include the Artificial Dynamics [Hig85] and the Resample-Move method [Sto02] [Fea02] [KBF+06].

2.2.3 Topology Estimation

The offline and online methodologies that were previously discussed to estimate the parameters of the HMMs maintain a fixed number of states throughout the estimation process. These methodologies are highly dependent on the number of states that are defined at the start of the learning process. The incorrect estimation of the number of states of the model has the following problems:

1. If the topology of the model is made too complex by having more states than required for a given sequence of observations, then some computational resources are going to be unnecessarily allocated during the parameter estimation process. This is not the only danger, complex models can also produce poor estimation results due to high variance [VEJ97].

2. When the topology of the model is made too simple, without enough states for a given sequence of the observations, the model will not capture the correct probability distributions for the sequence of observations [Sch02]. Notice that if two models of different complexity

are statistically equivalent, the simplest model is preferable due its efficiency in terms of computational resources.

The fact that the quality of the model is conditioned by the number of states of the model gives enough justification for the estimation of the best possible topology of the model given a sequence of observations.

There are some techniques that are able to find the best model, among a finite set of models, for a particular sequence of observations like the *Akaike Information Criterion* (AIC) and the *Bayesian Information Criteron* (BIC) [MZ97]. These criteria penalize the possible over fittings introduced by an excessive number of states. The problem with these criteria is that they can only be used after several models have already been estimated for a given sequence.

Some techniques allow the estimation of the topology of the model to occur in parallel with the estimation of the parameters. These techniques usually employ two basic strategies: *state splitting* or *state pruning* techniques. State splitting techniques start up with a very simple model that is incrementally turned more complex when certain criteria are met. State pruning techniques instead, start with a complex model that is successively simplified by the removal of the unnecessary states.

The work from Stenger et al. [SRP+01] performs incremental learning of the parameters and topology of the model by applying a splitting approach with a posterior validation of the change in the topology. The validation was done via goodness-of-fit tests like the χ^2 test and the Kolmogorov-Smirnov test.

The DISSOLVE algorithm from Vasko [VEJ97] is one batch topology learning algorithm that employs the state pruning approach. Vasko also proposed several topology estimators that include grammatical inference, *ad hoc* decomposition and information-theoretic approaches. Florez-Larrahondo [FL05] explored several such techniques and developed an incremental topology algorithm in conjunction with a incremental parameter estimation algorithm named tpIBW. This algorithm combines a parameter estimation algorithm, via IBW or IBW+ algorithms (section 2.2.2), with procedures that modify the topology of the model. The possibilities include removal of the transition with low probabilities and the removal of states by greedy or controlled pruning.

There are other approaches that do topology inference and structure learning of Bayesian networks. For example, Brand [Bra99] presents a mathematical framework for estimating parameters and simplifying the structure of the models with latent parameters that is able to prune states in times comparable to the conventional EM.

The incremental algorithms found in the literature that are designed for topology adaptation only allow the number of states of the model to increase or to decrease. This dissertation tries to fill this gap by devising an online algorithm capable of two way adaptation of the topology (section 4), that allow the number of states of the model to increase, decrease or maintain according with the observations.

Chapter 3

QHMM

This chapter discusses the framework of Quantitization-based Hidden Markov Models in more detail. This framework has some intrinsic properties that are interesting. These considerations are presented in a comparison with Baum-Welch algorithm in section 3.1.

Nevertheless, there was room for improvement. Section 3.2 elaborates all the improvements that were made to QHMM to extend the scope of their application and section 3.3 proceeds with the exposure of some of the results.

The improvements made to QHMMs described in this chapter together with the ideas of an online clustering algorithm, named ClusTrel, culminated in the realization of ClusTrelHMM, a novel algorithm presented in chapter 4.

3.1 Baum-Welch Comparison

Just by looking at the simplicity of the QHMM approach (section 2.2.1.2) we could predict it being faster and more efficient than the BW algorithm. The literature review confirms that the computational complexity of QHMM for parameter estimation is just $O(N^2)$ while the memory and time complexity of BW algorithm is $O(N^2T)$ per passage. The fact that BW algorithm sometimes requires many passages until convergence makes this matter even worse for the BW procedure.

Another obvious advantage that QHMM has over BW is that the emission distributions of the states are never redundant. In the BW case, usually when the number of states increases, some emission distributions of the states overlap near their the means. This is not case with QHMM because the partition of the space of the observations makes this phenomena impossible.

After implementing the basic QHMM procedure on R and comparing it to an existing implementation of the BW algorithm ('RHmm' package), preliminary results showed that the BW algorithm tends to be slower, as expected from above. But the BW algorithm was shown to have a tendency to fit the data sequences better than QHMM did for the same number of states. However, because QHMMs are faster, they can trade their time efficiency for better parameter estimation by an increment of the number of bins/states.

This first digression on code also allowed the understanding of other advantages of the QHMM procedure over the BW algorithm.

The BW algorithm is highly dependent on the initial parameters and so these parameters usually are initialized with some random seed. Consequently, they output different results for each run. The QHMM procedure, however, reliably outputs the same results over and over.

The BW procedure also has a tendency to need longer sequences of data for its operation because when the number of states increases, the number of sequences that are modeled tends to decrease. In contrast, the QHMM procedure is able to adjust the parameters of the model even with very short sequences.

3.2 Improvements

Despite the advantages of QHMMs considered in the previous section, we identify in this section some limitations of QHMM and show how the method can be improved. These improvements correspond to a better partition of the observation space, an extension to online operation and a topology adaptation.

3.2.1 Observation Space Partition

Although QHMM procedure looks at all the observations to compute the intervals of the bins, in the moment of dividing the space of the observations, it does not make use of this knowledge. The problem is that all the bins have the same size and therefore the same resolution. So, some bins may end up representing a huge number of observations while others barely none.

Another problem that arises with the fixed partition of the observation space is that sometimes the limits between the bins may end up dividing a certain distribution of the data unnaturally. For example, a certain normal distribution may end up being divided in two because of the definition of the limits of the bins.

To improve the partition of the space of observations, other methods like *equal-depth binning* and clustering through *k-means* were tried with good results, but other methods like Gaussian mixtures could have been used.

3.2.2 Online Operation

The QHMM procedure was adapted to work online after the topology of the model and their emission distributions were defined. This allowed the parameters to be updated after each new observation is available.

The trick was to notice that the sums of equation 2.19 can be stored and updated with each new observation. The only requirement is the definition of the sum of transition probabilities $STP(i, j)$ from state i to state j and the sum of probabilities $SP(i)$ of being in state i as separate variables

(equations 3.1 and 3.2). All $STP(i,j)$ can be condensed in a matrix STP of size $N \times N$ and all $SP(i)$ in a vector SP of size N.

$$STP(i,j) = \sum_{t=1}^{T-1} P(O_t|s_t = i)P(O_{t+1}|s_{t+1} = j) \tag{3.1}$$

$$SP(i) = \sum_{t=1}^{T-1} P(O_t|s_t = i) \tag{3.2}$$

The probabilities STP and SP are then used to update the transition matrix and the initial probabilities of the model with equations 3.3 and 3.4 instead of equations 2.19 and 2.20. After the application of previous equations the rows of the transition matrix and the initial probabilities vector are normalized.

$$a_{ij} = \frac{STP(i,j)}{SP(i)} \tag{3.3}$$

$$\pi_i = \frac{SP(i) + P(O_T|s_T = i)}{T} \tag{3.4}$$

Notice that the proposed improvement for the resolution of the space of the observations and the online extension of QHMM are not necessarily mutually exclusive.

The only requirement is that the partition of the space has also to occur online. Since clustering was an option for improving the partition of the observation space offline, it is obvious that any online clustering algorithm could do the same.

The problem in using an online clustering algorithm to improve the partition of the observation space is that it relaxes the QHMM condition of maintaining the intervals of the bins fixed throughout the process. As a consequence the emission distributions are always being updated and so the probabilities $P(O_t|s_t)$ are always computed based on different distributions.

In a nutshell, the online procedure provided by equations 3.1, 3.2, 3.3 and 3.4 provides equivalent results to the offline procedure present in the literature.

In practice, if an online clustering algorithm is used to improve and adapt the partition of the observation space online, those equations provide only an approximation of the results of the normal QHMM procedure because the means of the emission distributions are going to change during the process.

3.2.3 Topology Reconfiguration

The proposed online procedure for QHMMs also has the nice property of allowing topology adjustments and still provides good estimates for the parameters of the model after each adjustment.

The adjustment of the topology requires the resizing of:

- the vector with the probabilities of being in each state SP and
- the matrix with the sums of the transition probabilities STP.

After that, the procedure can follow normally as in the non-topology reconfiguration case.

In the case of adding a new state n to the model, a zero is added to vector SP as the state n has never been visited before (equation 3.5) and a row and a column are inserted in STP as no transitions from or to state n have occurred yet (3.6 and 3.7).

$$SP(n) = 0 \tag{3.5}$$
$$STP(i,n) = 0 \qquad 1 \leq i \leq N \tag{3.6}$$
$$STP(k,n) = 0 \qquad 1 \leq i \leq N \tag{3.7}$$

It is also possible to decrease the number of states of the HMM by merging the states through the sum. In the case of merging states a and b in a state m, the sum probabilities of being in state m is equal to the sum of the probabilities of being in states a and b (equation 3.8). Similarly, the rows a and b and columns a and b of STP have to be summed has described by equations 3.9 and 3.10. Lastly the sums of the probabilities of transitioning from m to itself equals the sum of probabilities of transitioning from state a to state a, plus the sums of probabilities of transitioning from state b to state b and the sum of probabilities of transitioning from a to b and b to a (equation 3.11).

$$SP(m) = SP(a) + SP(b) \tag{3.8}$$
$$STP(m,i) = STP(a,i) + STP(b,i) \qquad 1 \leq i \leq N \tag{3.9}$$
$$STP(i,m) = STP(i,a) + STP(i,a) \qquad 1 \leq i \leq N \tag{3.10}$$
$$STP(m,m) = STP(a,a) + STP(b,b) + STP(a,b) + STP(b,a) \tag{3.11}$$

3.3 Results

The basic QHMM procedure and all the improvements discussed in the previous sections were implemented using the R programming language.

This section presents two examples of the application of the adaptation of the topology for two sequences of data sequences of data.

In the end of this section, it will be easier to grasp why QHMMs are particularly suitable to model even small sequences of data.

3.3.1 Online Topology Adaptation

The purpose of this section is to show how the topology of the model can easily be adapted when desirable. Since the emission distributions are defined a priori, only the evolution of the transition matrix is going to be analyzed.

The first example will deal with an insertion of a state into a 2-state model while trying to learn the parameters for the following sequence:

$$10, 20, 10, 20, 30, 30$$

The model starts with the means equal to $(10, 20)$ and standard deviations equal to $(1, 1)$. After the fourth observation the model is resized to include a new state with mean 30 and standard deviation equal to one. The evolution of the transition matrix is presented next. The QHMM procedure only updates the model after the first two observations.

$$\begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{pmatrix} \xrightarrow{10,20} \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} \xrightarrow{10} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \xrightarrow{20} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \xrightarrow[insertion]{state} \xrightarrow[insertion]{state} \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0.33 & 0.33 & 0.33 \end{pmatrix} \xrightarrow{30} \begin{pmatrix} 0 & 1 & 0 \\ 0.5 & 0 & 0.5 \\ 0.33 & 0.33 & 0.33 \end{pmatrix} \xrightarrow{30} \begin{pmatrix} 0 & 1 & 0 \\ 0.5 & 0 & 0.5 \\ 0 & 0 & 1 \end{pmatrix}$$

The logic is that the QHMM procedure always looks at the last two observations and computes the probabilities of transitioning between every pair of states at that moment in time. Those probabilities are going to contribute for the elements of the transition matrix as followed by equation 2.19. Some of the values of the transition matrices may not be intuitive because the rows of the transition matrix are normalized.

The second example shows how two states in a 3-state model are merged while estimating the parameters of the following sequence:

$$10, 10, 20, 20, 30, 30$$

The model starts with the means equal to $(10, 20, 30)$ and standard deviations equal to $(1, 1, 1)$ and merges states that correspond to the means $(20, 30)$ in the end of all observations of the sequence.

$$\begin{pmatrix} 0.33 & 0.33 & 0.33 \\ 0.33 & 0.33 & 0.33 \\ 0.33 & 0.33 & 0.33 \end{pmatrix} \xrightarrow{10,10} \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \xrightarrow{20} \begin{pmatrix} 0.5 & 0.5 & 0 \\ 0.5 & 0.5 & 0 \\ 0.5 & 0.5 & 0 \end{pmatrix} \xrightarrow{20} \begin{pmatrix} 0.5 & 0.5 & 0 \\ 0 & 1 & 0 \\ 0.33 & 0.66 & 0 \end{pmatrix} \xrightarrow{30} \xrightarrow{30} \begin{pmatrix} 0.5 & 0.5 & 0 \\ 0 & 0.5 & 0.5 \\ 0.25 & 0.5 & 0.25 \end{pmatrix} \xrightarrow{30} \begin{pmatrix} 0.5 & 0.5 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & 0 & 1 \end{pmatrix} \xrightarrow[states]{merge} \begin{pmatrix} 0.5 & 0.5 \\ 0 & 1 \end{pmatrix}$$

The important aspect to retain here is that when the model is adjusted in the last two examples, by increasing or decreasing the number of states, the model makes use of all information that was gained by the past observations.

Chapter 4

ClusTrelHMM

This chapter describes a novel algorithm for estimating the parameters of Gaussian HMMs while allowing the number of states of the model to change with incoming observations. The proposed algorithm is an adaptation of the ClusTrel algorithm and because of this a small introduction to ClusTrel is going to be made before proceeding to the full explanation of the proposed algorithm.

4.1 ClusTrel

The ClusTrel algorithm [MM12] is an online clustering algorithm able to process data streams and select the number of clusters that best represent the stream according to a clustering evaluation index index.

This algorithm stores and maintains in parallel N clustering structures representing N clustering possibilities of the stream with $1,...,N$ clusters. All these clustering structures are then updated dynamically based on the Viterbi [Rab89][Bis06, chap. 13] algorithm each time a new point is available.

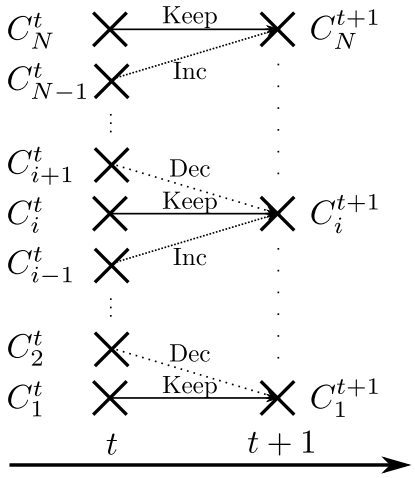

Figure 4.1: ClusTrel's trellis structure.

The clustering structure C_i^t represents one clustering possibility of t points and i number of clusters. This clustering structure is nothing more than a compact representation of the clustered data. Each clustering structure is a tuple (N, C, SSQ) where:

- N is a vector with the number of points in each cluster;
- C is a vector with the centroids of each cluster;
- and SSQ is a vector with the sum of squared distances of the points in each cluster to its centroid.

ClusTrel does not update all clustering structures individually because this would restrict severely the number of the clustering possibilities. Instead, ClusTrel computes C_i^{t+1}, the clustering structure at time $t+1$ with i clusters, as shown in figure 4.1, from three possibilities: by increasing the number of states of C_{i-1}^t (*Inc* transition), by maintaining the number of states of C_i^t (*Keep* transition) or by decreasing the number of states C_{i+1}^t (*Dec* transition). The selected clustering structure is the one that minimizes the total SSQ of the clustering structure.

In figure 4.1 we can see that the clustering structures at the top and at the bottom of the trellis (C_N^{t+1} and C_1^{t+1}) can only be updated from two possibilities because ClusTrel is only concerned with clustering with $1, ..., N$ clusters.

In a nutshell, *Inc* transitions are characterized by the addition of a new point as a new centroid, *Keep* transitions add the new point to the closest cluster and *Dec* transitions merge the two closest clusters and add the new point to the closest cluster of the result.

The best clustering structure at any given time can be selected amongst the N clustering structures using a cluster evaluation index named *MDB index* [MM12]. This has the advantage over the classical *DB index* of being computable online [MM12]. The MDB index for each clustering structure is defined as follows:

$$MDB = \frac{1}{N} \sum_{i=1}^{N} \max_{i \neq j} \frac{SSQ(C_i) + SSQ(C_j)}{dist(C_i, C_j)^2} \tag{4.1}$$

where $SSQ(C_i)$ represents the average sum of squares distances inside cluster with centroid C_i and $dist(C_i, C_j)$ the distances between the centroids C_i and C_j.

4.2 ClusTrelHMM

ClusTrelHMM is an extension of the ClusTrel algorithm for the estimation of the parameters of Gaussian HMMs. It extends the concept of the trellis structure of ClusTrel to Hidden Markov Models and uses the clustering structures of ClusTrel to compute the means and standard deviations of the emission distributions.

In this way, the proposed method maintains in parallel N HMMs with $1, ..., N$ states with the goal of finding the best one that describes the sequence of observations. This algorithm enables the detection of possible changes in concept of the observations.

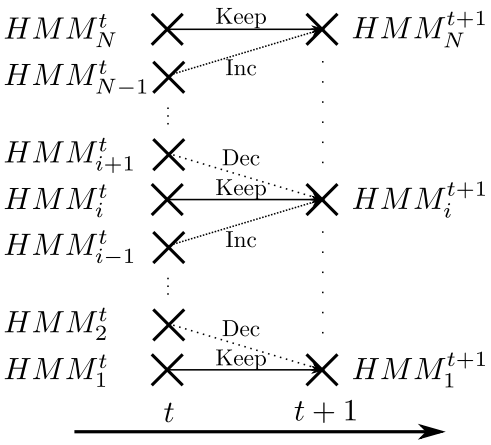

Figure 4.2: ClusTrelHMM's trellis structure.

Internally, ClusTrelHMM maintains N clustering feature vectors and updates them in the same way as in ClusTrel.

The emission distributions of the HMMs are computed using the clustering structures updated by the ClusTrel procedures. Each clustering structure with n number of clusters is responsible for updating the emission distributions of the HMM with n number of states. The means of the distributions of each state are given by the correspondent centroid C_i in the clustering structure (equation 4.2). The standard deviations of the distributions are equal to the square root of the average sum of squared distances of the correspondent clustering structure (equation 4.3).

$$\mu_i = C_i \tag{4.2}$$

$$\sigma_i = \sqrt{\frac{SSQ_i}{N_i}} \tag{4.3}$$

The complete definition of each HMM present in the trellis also requires to define the transition matrix and the initial probabilities. These probabilistic distributions are calculated based on the online adaptation of the QHMM procedure discussed in section 3.2.2. The transition matrices are calculated using equation 3.3, while the initial probabilities are computed through equation 3.4.

If the relationship between the past and current HMM structures present in the trellis occurs through an *Inc* or *Dec* transition, all the data structures related to the HMM parameter estimation have to be resized before being updated. This includes the resizing of the QHMM structures and the resizing of HMM parameters, which includes the transition matrix, the initial probabilities vector, the means vectors and the standard deviations vector.

The resizing of the QHMM procedure corresponds to the improvement of the topology adaptation of the QHMM procedure described in section 3.2.3.

The transition matrices and the initial probabilities also have to be resized conveniently before being updated by the adapted QHMM procedure.

4.3 Results

After the design and implementation of ClusTrelHMM algorithm in the R language, its behavior was analyzed with the goal of understanding its advantages and potential pitfalls. The performance of ClusTrelHMM was tested against both synthetic and real network data.

An analysis of how the parameters are estimated is provided in section 4.3.1, while the analysis of the topology adaptation is done in section 4.3.2.

The overall performance of the algorithm was tested against a huge variety of data sequences provided by a real DSL dataset. This analysis is presented in section 4.3.3.

4.3.1 Parameter Estimation

The analysis of the evolution of the estimated parameters from ClusTrelHMM required the generation of synthetic data sequences to be able to compare the estimated parameters against the parameters that were used in the generation of the synthetic sequences. This tailored approach allowed us to demonstrate specific behaviors of the algorithm that would otherwise be harder to understand.

The notation $S_i = G(L, N, \pi, A, \mu, \sigma)$ represents a sequence S_i of length L that was generated from an Gaussian HMM with N states, with initial probabilities π, with a transition matrix A and emission distributions with means μ and standard deviation σ. The example sequences that were used to demonstrate ClusTrelHMM behavior are defined below.

$$S_1 = G\left(L = 1000, N = 2, \pi = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, A = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}, \mu = \begin{bmatrix} 10 \\ 20 \end{bmatrix}, \sigma = \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)$$

$$S_2 = G\left(L = 1000, N = 2, \pi = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, A = \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix}, \mu = \begin{bmatrix} 10 \\ 20 \end{bmatrix}, \sigma = \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)$$

$$S_3 = G\left(L = 1000, N = 2, \pi = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, A = \begin{bmatrix} 0.6 & 0.4 \\ 0.6 & 0.4 \end{bmatrix}, \mu = \begin{bmatrix} 10 \\ 20 \end{bmatrix}, \sigma = \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)$$

$$S_4 = G\left(L = 1000, N = 3, \pi = \begin{bmatrix} 0.33 \\ 0.33 \\ 0.33 \end{bmatrix}, A = \begin{bmatrix} 0.33 & 0.33 & 0.33 \\ 0.33 & 0.33 & 0.33 \\ 0.33 & 0.33 & 0.33 \end{bmatrix}, \mu = \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix}, \sigma = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}\right)$$

$$S_5 = G\left(L = 1000, N = 3, \pi = \begin{bmatrix} 0.33 \\ 0.33 \\ 0.33 \end{bmatrix}, A = \begin{bmatrix} 0.5 & 0.3 & 0.2 \\ 0.9 & 0 & 0.1 \\ 0.1 & 0.2 & 0.7 \end{bmatrix}, \mu = \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix}, \sigma = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}\right)$$

$$S_6 = G\left(L = 1000, N = 3, \pi = \begin{bmatrix} 0.33 \\ 0.33 \\ 0.33 \end{bmatrix}, A = \begin{bmatrix} 0 & 1 & 0 \\ 0.33 & 0.33 & 0.33 \\ 0.5 & 0.5 & 0 \end{bmatrix}, \mu = \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix}, \sigma = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}\right)$$

The evolution of the parameter estimation of HMMs for sequences $S_1 - S_5$ for ClusTrelHMM and IBW.

The first conclusion that should be made is that ClusTrel worked extremelly well in finding the means of the emission distributions in all the sequences (see figures 4.3, 4.5, 4.7, 4.9, 4.11 and 4.13). These results are striking because IBW did not manage to find the right means in sequences S_3, S_4, S_5 (figures 4.3, 4.9 and 4.11). IBW is highly dependent on the means that are found in the beginning of the estimation. If those means are far from the correct ones, the algorithm does not converge to the correct means or does it very slowly.

Surprisingly, the evolution of the parameters of ClusTrelHMM and IBW were very similar for sequences S_1, S_2 and S_6 (figures 4.3, 4.4, 4.5, 4.6, 4.13 and 4.14). The similarity of ClusTrelHMM and IBW for these sequences is also reflected in terms of log-likelihoods. These similarities result from both algorithms working incrementally and computing the same probabilities of being in each state at each moment in time.

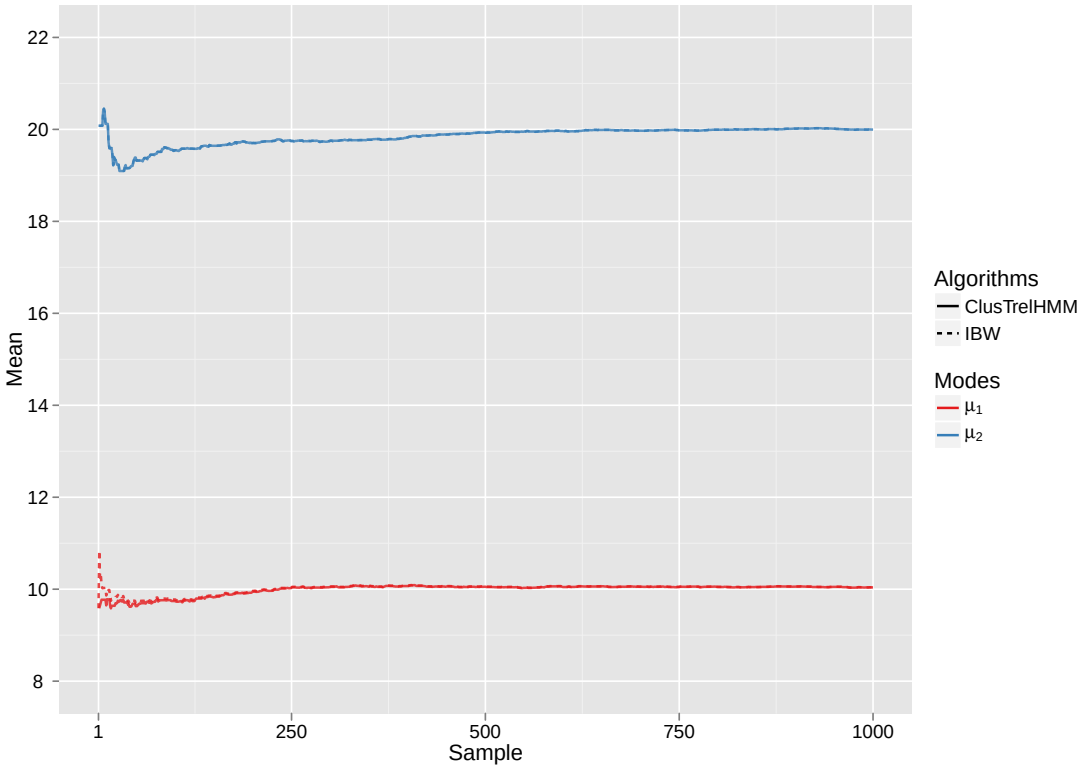

Figure 4.3: Evolution of the means of the emission distributions for sequence S_1.

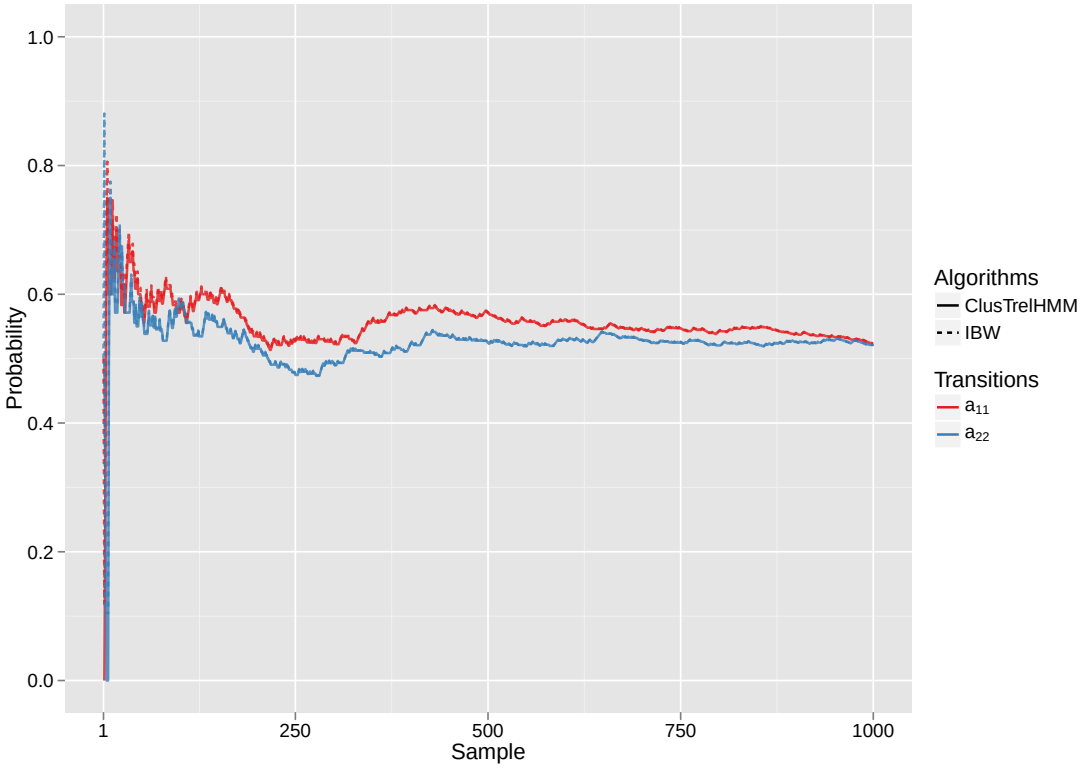

Figure 4.4: Evolution of the transition probabilities for sequence S_1.

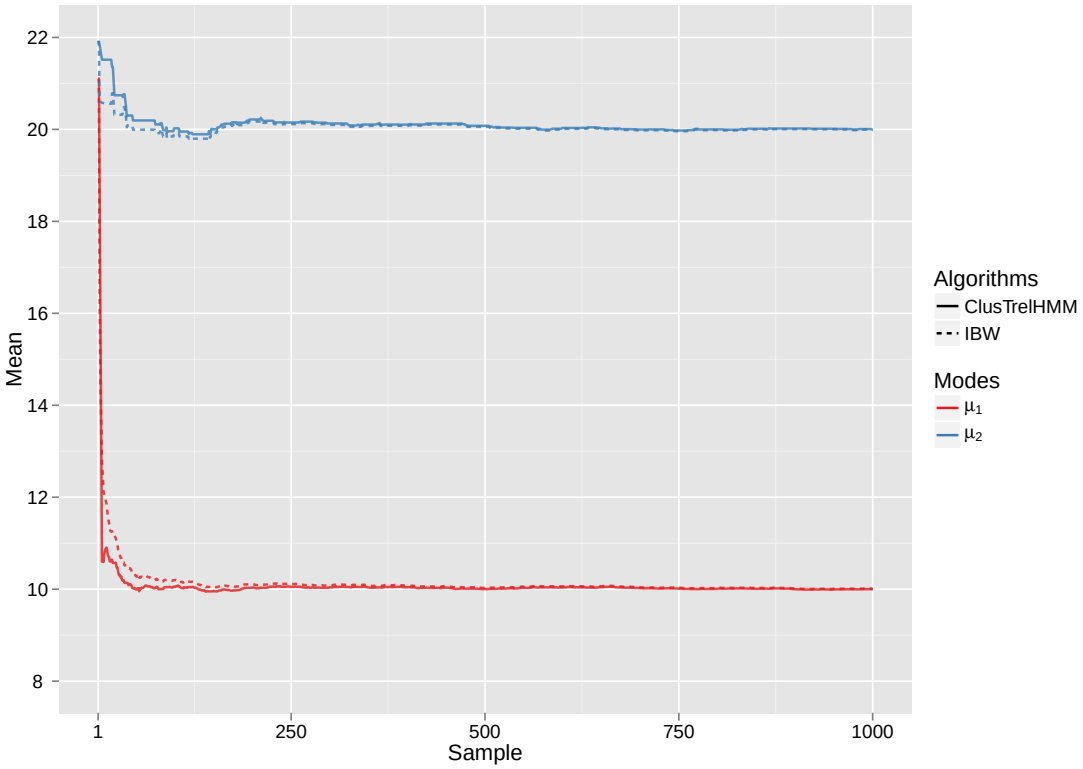

Figure 4.5: Evolution of the means of the emission distributions for sequence S_2.

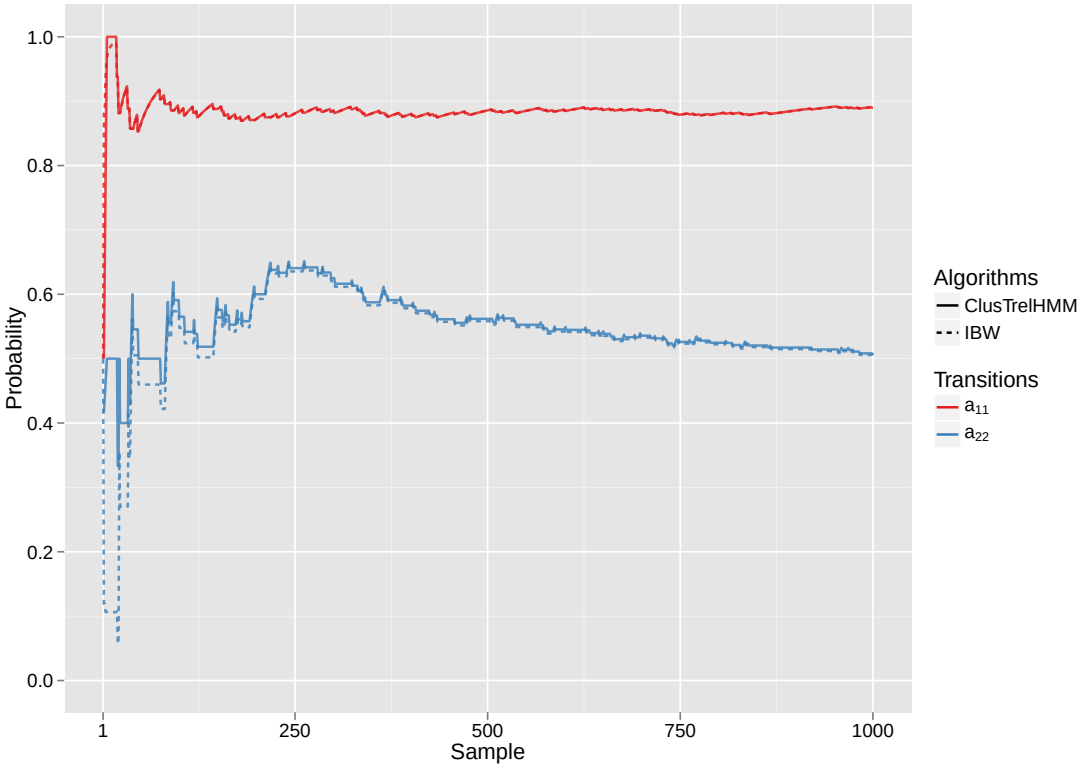

Figure 4.6: Evolution of the transition probabilities for sequence S_2.

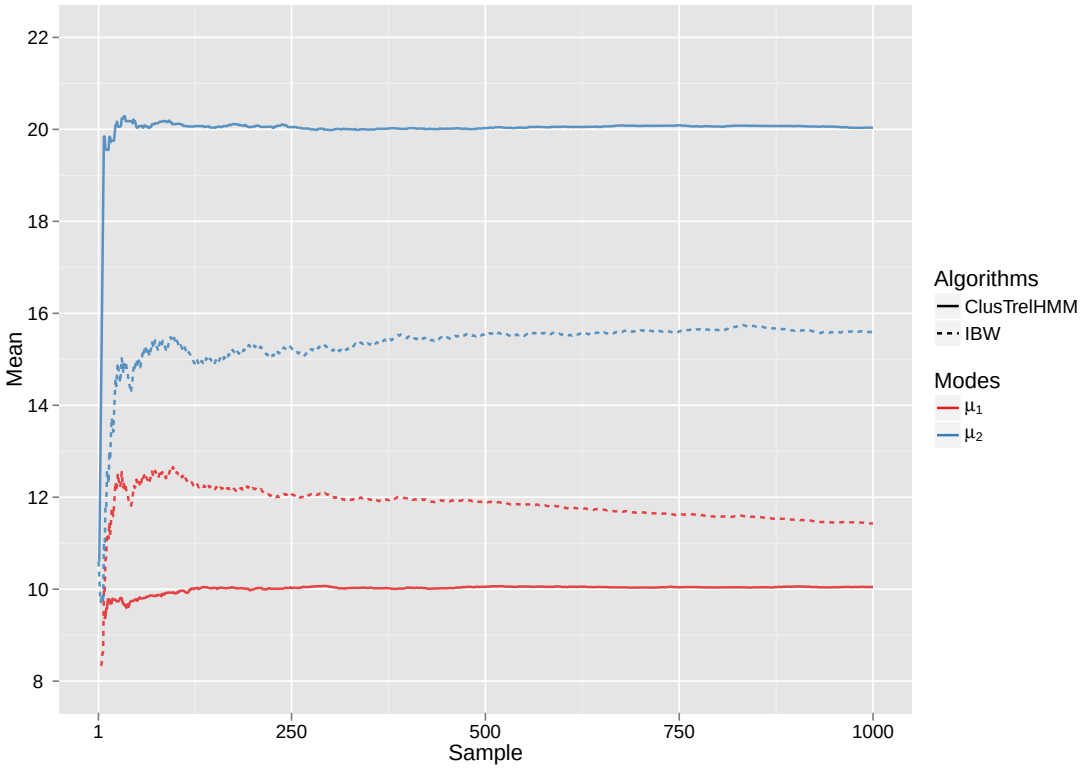

Figure 4.7: Evolution of the means of the emission distributions for sequence S_3.

Figure 4.8: Evolution of the transition probabilities for sequence S_3.

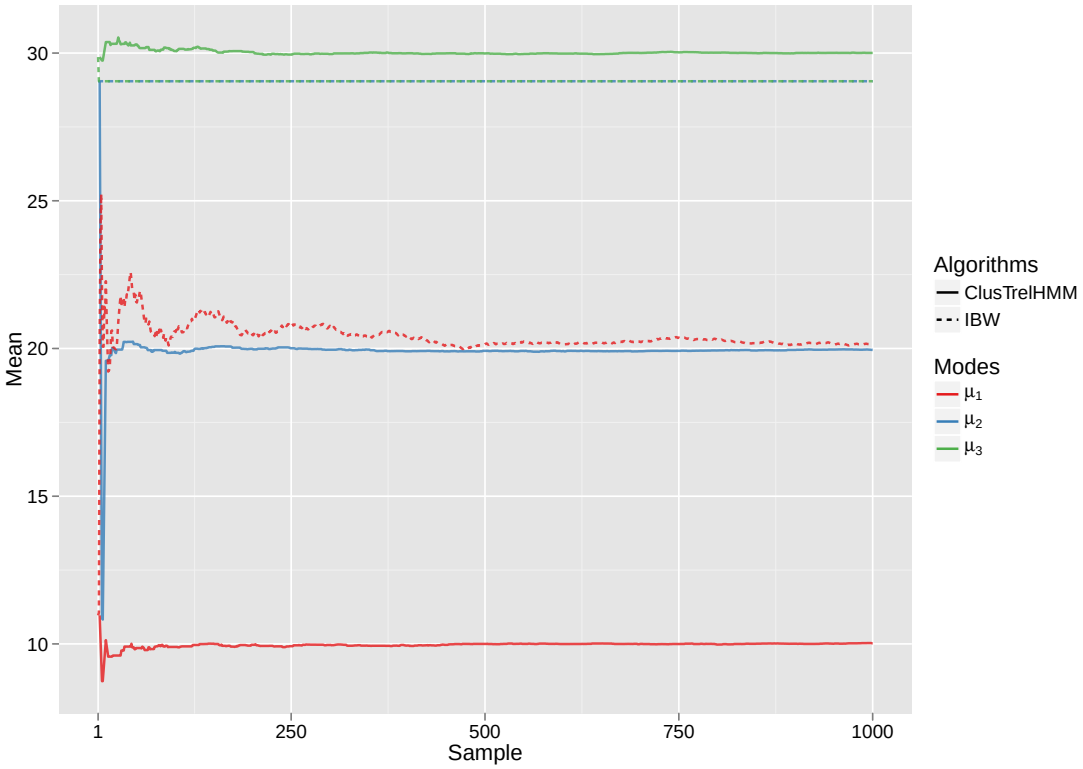

Figure 4.9: Evolution of the means of the emission distributions for sequence S_4.

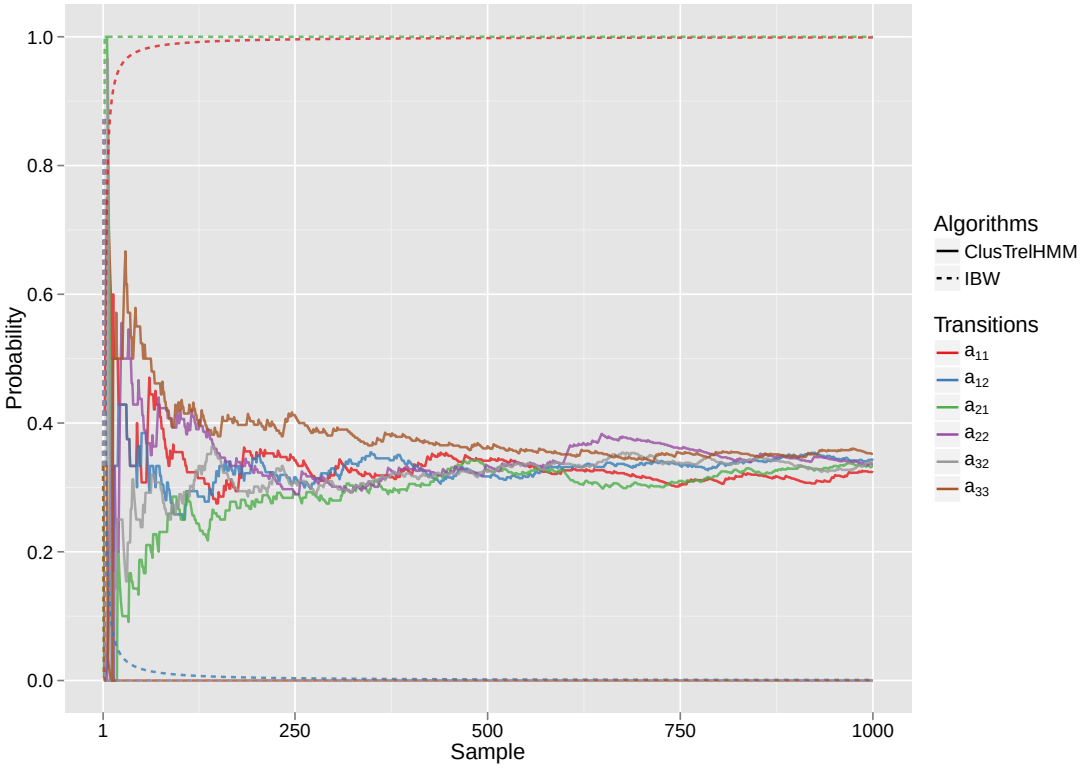

Figure 4.10: Evolution of the transition probabilities for sequence S_4.

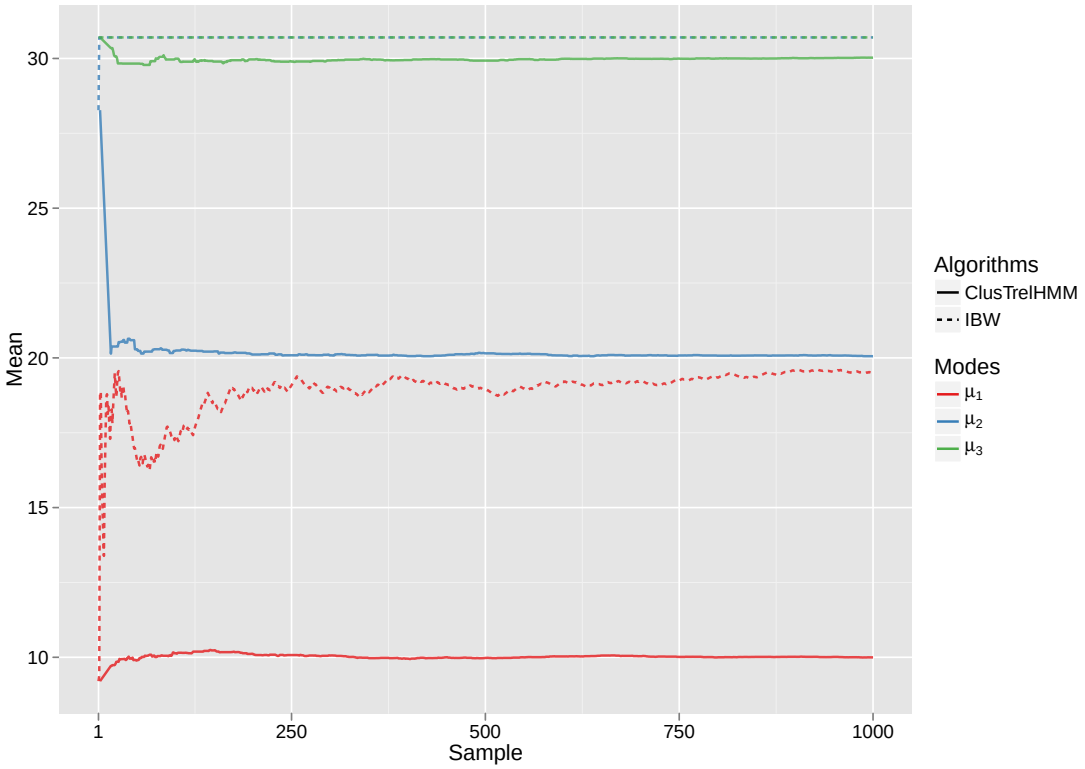

Figure 4.11: Evolution of the means of the emission distributions for sequence S_5.

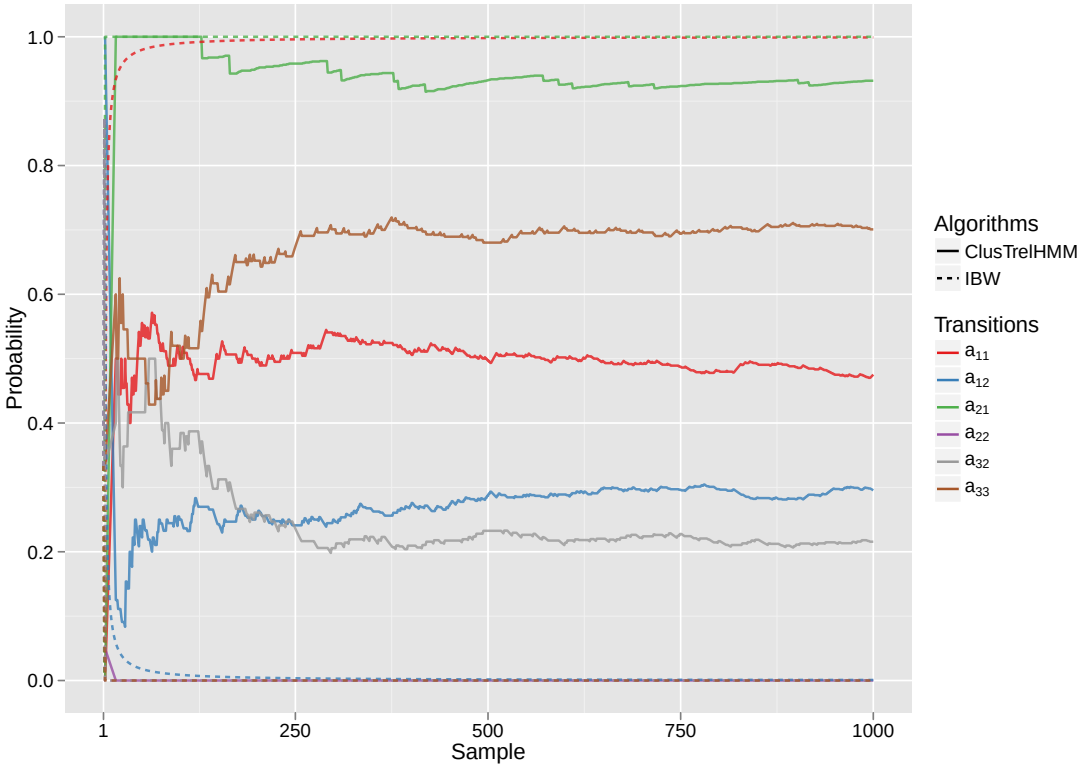

Figure 4.12: Evolution of the transition probabilities for sequence S_5.

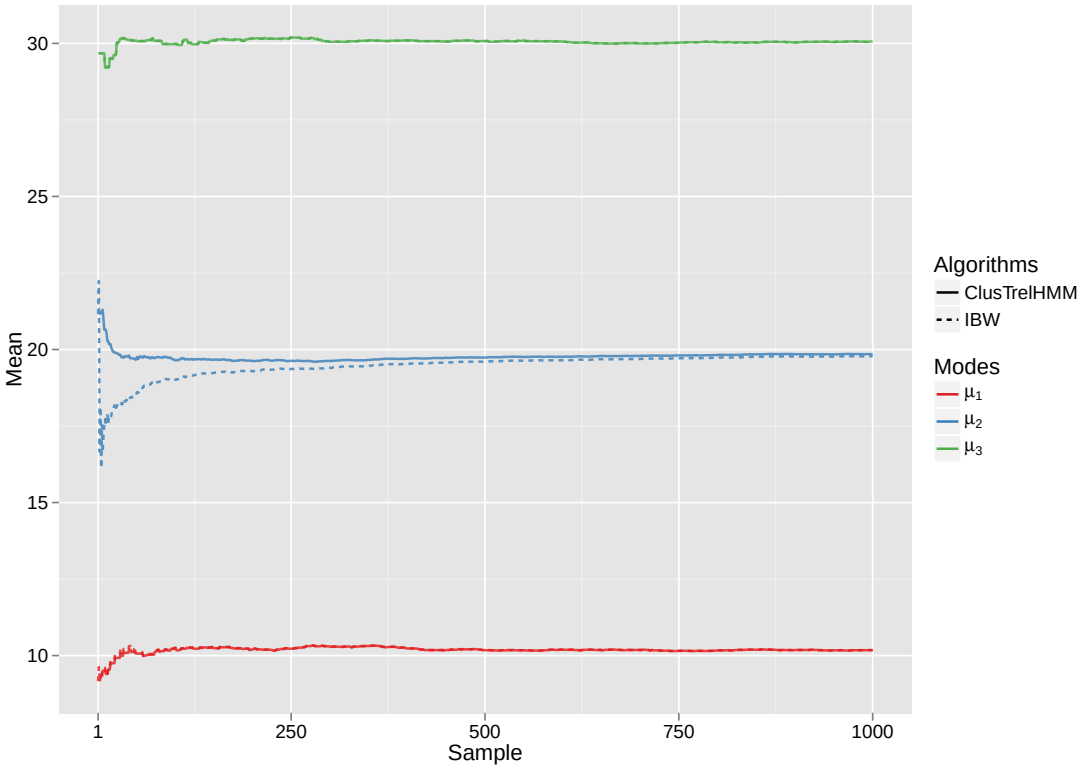

Figure 4.13: Evolution of the means of the emission distributions for sequence S_6.

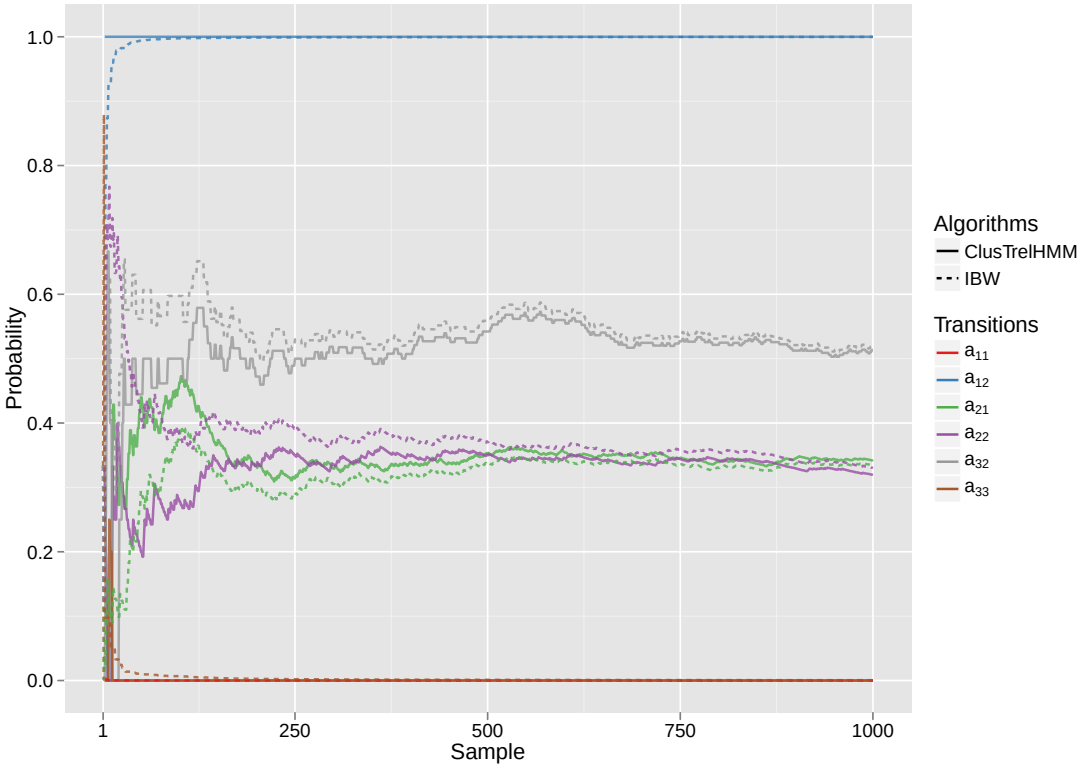

Figure 4.14: Evolution of the transition probabilities for sequence S_6.

4.3.2 Topology Adaptation

This section investigates the topology adaptation of ClusTrelHMM. For that purpose, two sequences of synthetic data were generatered with two different modes, each using HMMs with 2 and 3 states. All HMMs that were used to generate these synthetic sequences had unitary standard deviations, equiprobable transition probabilities for each pair of states and equiprobable initial probabilities.

The synthetic sequences S_7 and S_8 are depicted by the graphs of the figures 4.15 and 4.16. The first sequence served to analyze an increment in the number of states of the model because the sequence starts with two modes (10 and 30) and finishes with three modes (10, 20, 30). The second sequence, on the other hand, starts with three modes (10, 20, 30) and finishes with only two modes (10, 25) and allowed the analysis of a decrement in the number of states of the model.

The evolutions of the best number of states of ClusTrelHMM computed for sequences S_7 and S_8 are presented in figures 4.17 and 4.18 respectively. In both figures, the number of states was overestimated in the beginning of each sequence because ClusTrel tries to allocate eagerly the observations to the maximum possible number of states but this effect is quickly corrected.

ClusTrelHMM takes about 125 observations to detect the change in context that occurs at sample 1001 for sequence S_7 (figure 4.17). The delay in the adjustment is worse in sequence S_8, taking about 700 observations for the adjustment to happen (figure 4.18). The reason ClusTrelHMM made sequence S_8 lag behind S_7 in terms of the response to the change in context is that the new state with mean 25 is closer to the existent states with means 20 and 30.

Finally, the evolutions of the means for the emission distribution of the best model provided from ClusTrelHMM, for sequences S_7 and S_8, are presented in figures 4.19 and 4.20 where the delays in the detection are again visible.

In the case of sequence S_7, after the change in context that occurs in the middle of the sequence and before the detection of the change in context happens, the means of the existent states start to bend towards the mean of the new state. In the case of figure S_8, something similar occurs, the means of the states with modes defined at 20 and 30 start to bend towards the future new state that is the merging result of the other two.

Figure 4.15: Sequence S_7.

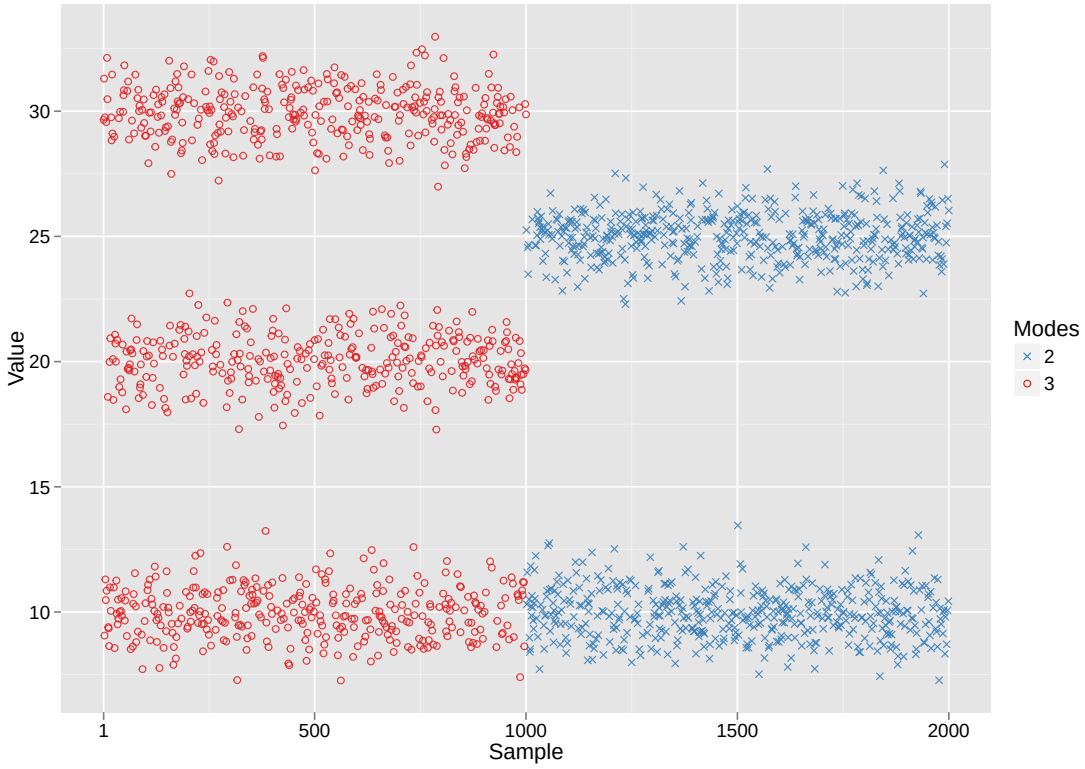

Figure 4.16: Sequence S_8.

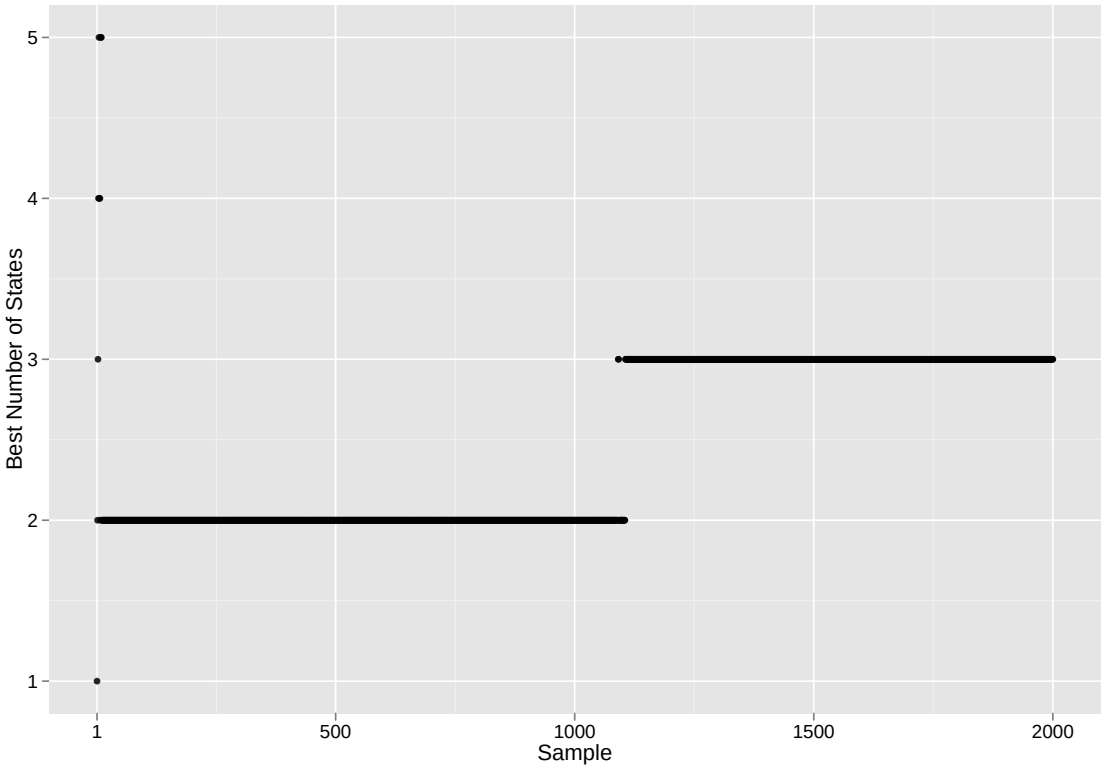

Figure 4.17: Evolution of the best number of states of ClusTrelHMM for sequence S_7.

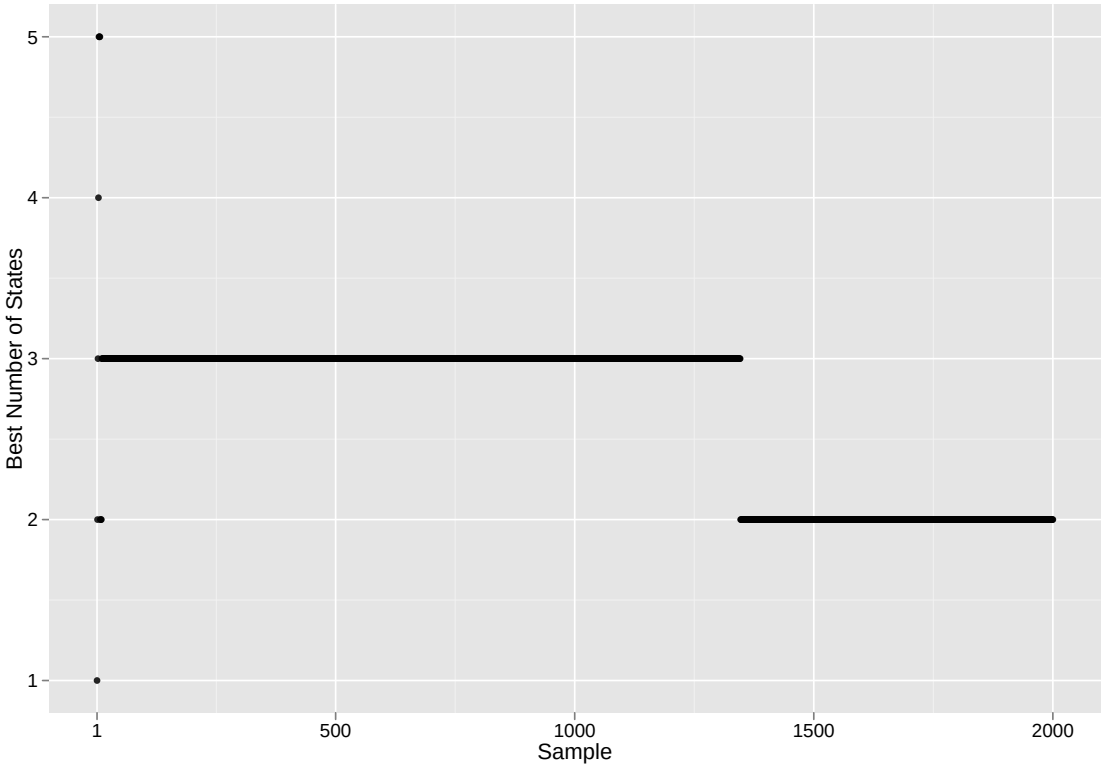

Figure 4.18: Evolution of the best number of states of ClusTrelHMM for sequence S_8.

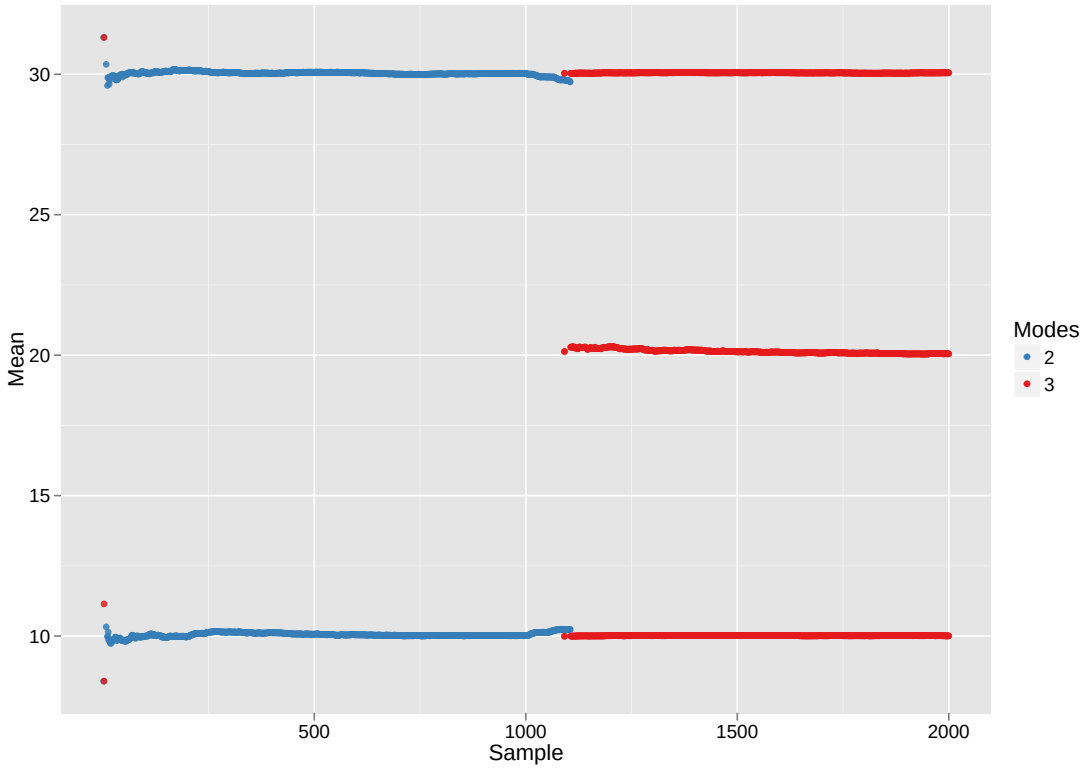

Figure 4.19: Evolution of the means for the best HMM for sequence S_7

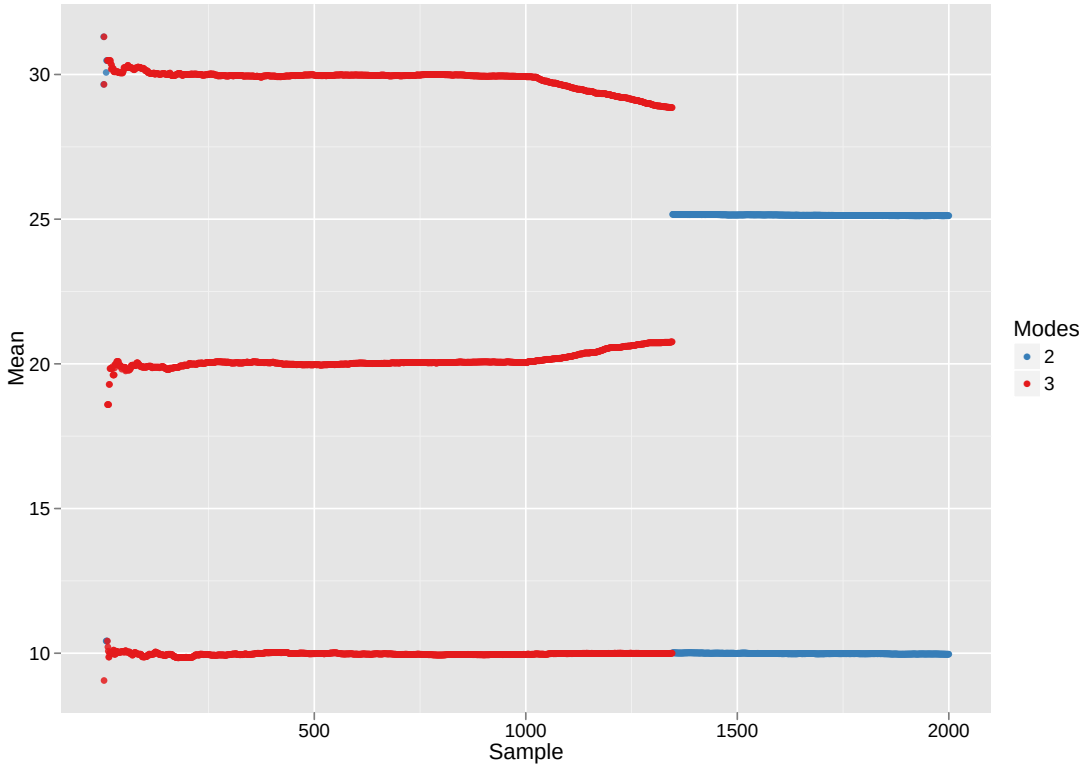

Figure 4.20: Evolution of the means for the best HMM for sequence S_8

4.3.3 Dataset

The available DSL network dataset is composed of 195026 sequences. These sequences are composed of the interarrival times between consecutive error messages for different customers in a DSL network, completing a total of 720 hours of communications. This dataset serves the purpose of shading some light upon the performance of the algorithms against a huge diversity of sequences that are only present in the systems of the real world.

The BW and IBW algorithms were chosen to serve as comparisons against ClusTrelHMM. The BW algorithm is an offline algorithm that usually produces very good estimates of the parameters of the model because it is able to pass several times through the data. A more fair assessment is made against IBW algorithm, which is an online algorithm that only passes one time through the data as the ClusTrelHMM algorithm does. Later in this section, we see that the estimation made using ClusTrelHMM is not too far from BW and better than IBW.

The procedure used for the BW algorithm is available in a R package named 'RHmm', but the IBW algorithm was implemented during the course of this work.

The overall score for the performance of each of the algorithms against the complete dataset was computed as follows. Each algorithm was applied to every sequence in the dataset, resulting in the estimation of a model per sequence in the dataset. The probabilities of the resulting models to generate the sequences for which they were adjusted was evaluated. Usually these likelihoods are presented in terms of log-likelihood to avoid possible underflows. The performance score of each algorithm was computed by averaging all the log-likelihoods of the entire dataset.

The previous process was repeated several times for models with different number of states. The average log-likelihood of the BW and IBW algorithms for the entire dataset was evaluated in 8 different simulations for models with 2, 3, 4 and 5 states. Only one simulation was executed for ClusTrelHMM with the maximum number of states as 5 but, a total of five average log-likehoods were calculated: four that correspond to the internal HMMs with 2, 3, 4 and 5 states and the other by choosing the selected model by ClusTrelHMM.

The results of the average log-likelihoods for each of the algorithms against the entire dataset are condensed in figure 4.21. The average log-likelihoods for all the internal HMMs of ClusTrelHMM have a tendency to be better than those provided by IBW for the same number of states. Contrary, all HMMs from ClusTrelHMM have a tendency to be worse than those provided by BW algorithm, which is normal because BW is an offline iterative algorithm.

Figure 4.21 also shows that the likelihoods for all the algorithms improved with the number of states which indicates that the sequences of dataset could have been modeled better with a higher number of states. Nevertheless, these simulations limited to a maximum of 5 states provided valuable information for the comparison of the algorithms.

The average log-likelihood for the selected HMM provided by ClusTrelHMM was 301, which again supports that ClusTrelHMM has a tendency to be better than IBW and worse than BW.

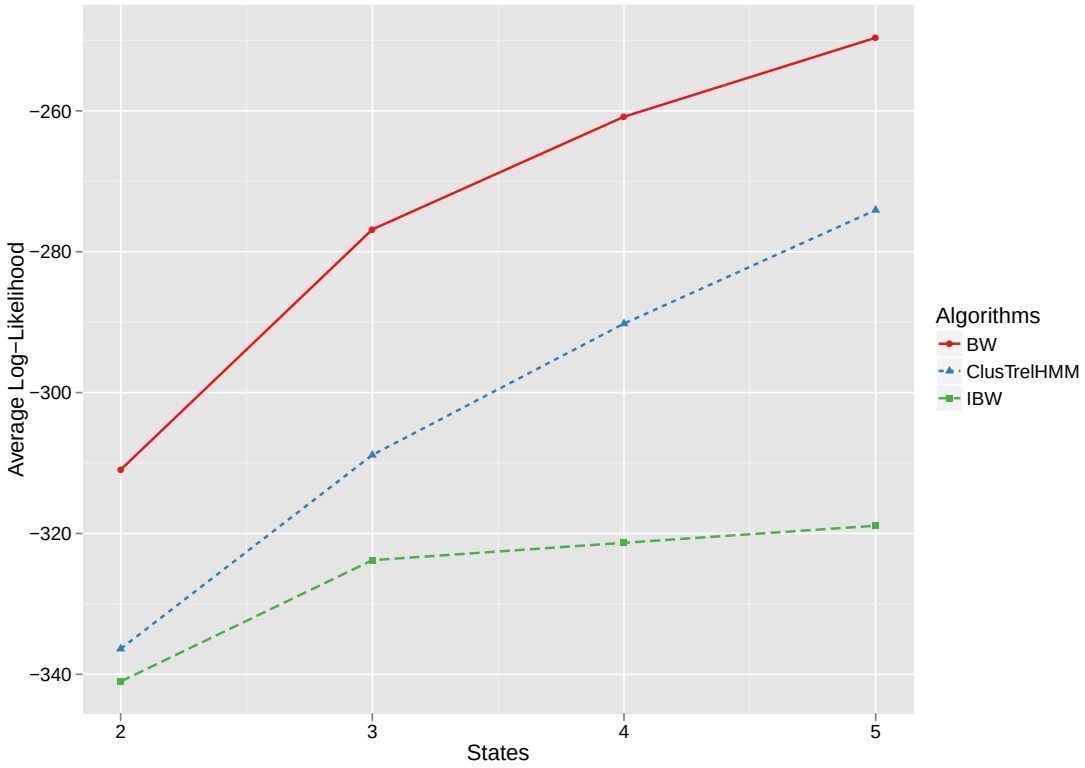

Figure 4.21: Average log-likelihoods for the entire dataset.

For better understanding of the differences in terms of log-likelihoods of ClusTrelHMM and IBW, the distributions of the log-likelihood differences for each of the sequences are plotted in figures 4.22 and 4.23.

Figure 4.22 plots the distribution of the log-likelihood differences between the models of 5 states of ClusTrelHMM with the IBW model of 5 states. Figure 4.23 plots the distribution of the log-likelihood differences for the selected model of ClusTrelHMM with the IBW model with the same number of states.

In both cases, positive values of log-likelihood difference means that ClusTrelHMM provides a better estimation of the parameters than IBW for that sequence and negative values represents the inverse.

In both plots, the great majority of the log-likelihood differences have absolute values with less than 250 and ClusTrelHMM tends to perform better than IBW. Numerically, figure 4.22 shows that 84% of the sequences were modeled better with ClusTrelHMM than IBW, while figure 4.23 shows that 76% of the sequences were modeled better with ClusTrelHMM.

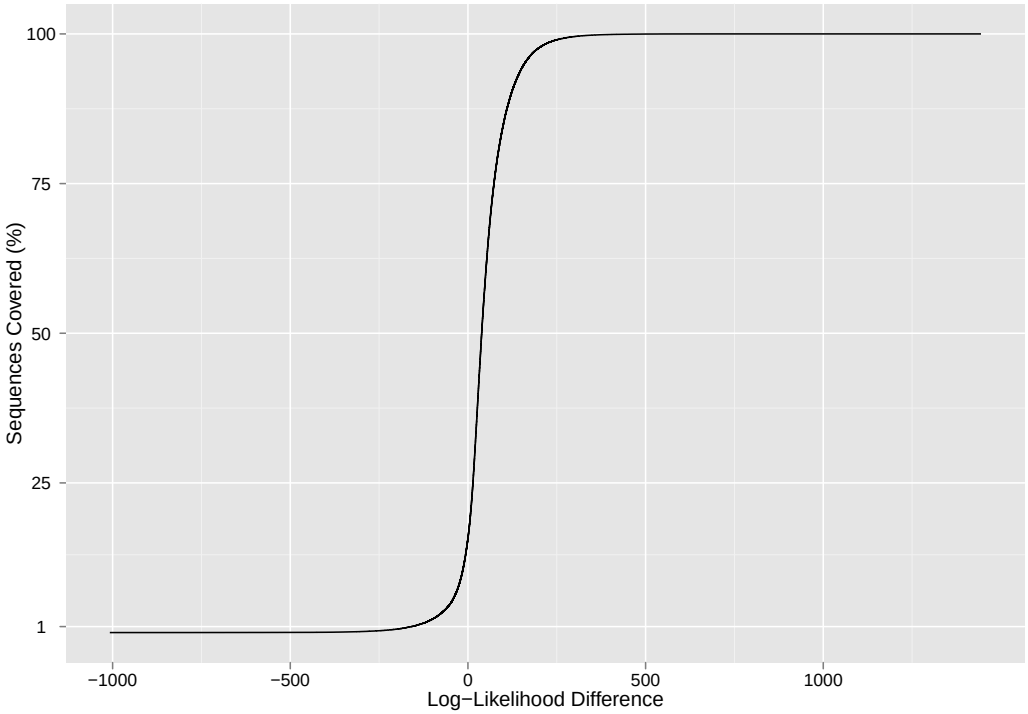

Figure 4.22: Distribution of the log-likelihoods difference between HMM with 5 states of ClusTrelHMM and IBW.

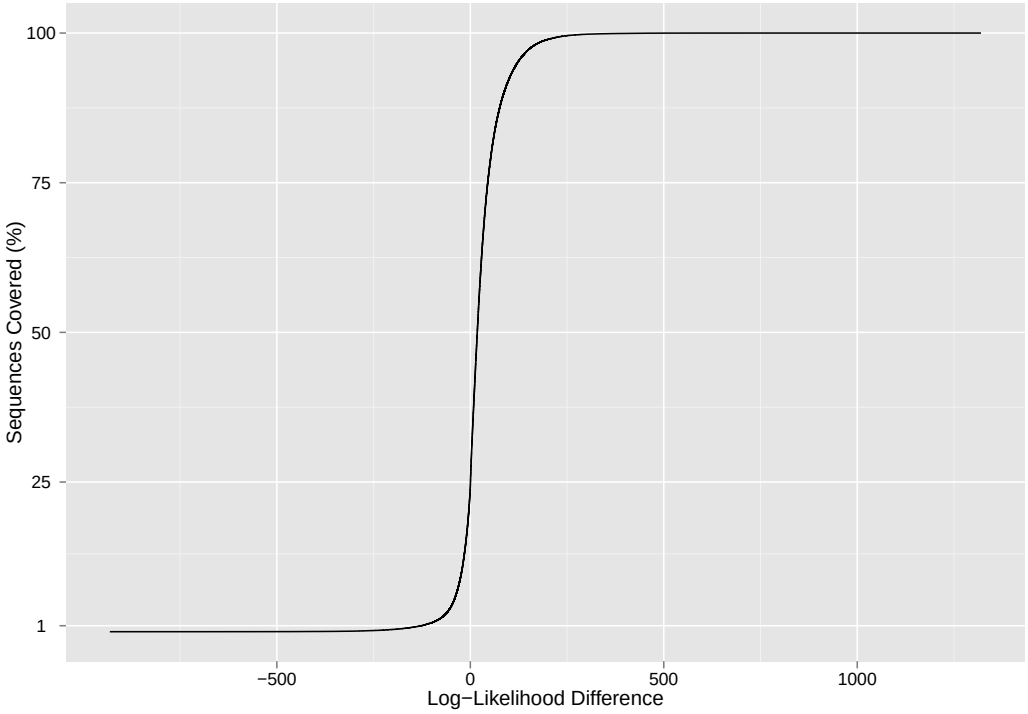

Figure 4.23: Distribution of the log-likelihoods difference between the selected HMM from ClusTrelHMM and the IBW model with the same number of states.

4.3.4 Conclusion

The results of the experiments done with ClusTrelHMM prove that this algorithm is a very solid alternative for the online estimation of the parameters of Hidden Markov Models.

The study of the evolution of the estimated parameters of subsection 4.3.1 allowed asserting that the algorithm is able to converge to the correct parameters, even when IBW algorithm does not managed to do the same.

The analysis of the topology adaptation of subsection 4.3.2 shows that the algorithm is able to choose the model with the number of states that best describes the observations.

The application of ClusTrelHMM to a large dataset of sequences of data presented in subsection 4.3.3 proved that the quality of the estimation, provided by the selected HMM of ClusTrelHMM, tended to be better than the ones provided by IBW. In fact, all internal HMMs of ClusTrelHMM have a tendency to outperform the models with the same number of states provided by the IBW algorithm.

Chapter 5

Conclusion

5.1 Conclusion

In the literature there is an uncountable number of ways to estimate the parameters of Hidden Markov Models. This dissertation tried to provide an overview of some algorithms that were the backbone of this field of study.

A very simple approach, called Quantization-based Hidden Markov Model, was found to be able to estimate the parameters offline with one fast sequential passage through the data. It was also noticed that this method could be improved in several ways. The partition of the observation space could be improved through clustering. The method could also be adapted to work incrementally and to support the reconfiguration of the topology of the model.

This dissertation focused on the improvements behind Quantization-based Hidden Markov Models. This lead to the creation of a novel online free topology algorithm for HMM parameter estimation which combines the improvement ideas of QHMM with an online clustering algorithm named ClusTrel.

The resulting algorithm was named ClusTrelHMM and is able to find the number of states of the model that best describes a given sequence. This facilitates the detection of the changes in the context of the observations.

The results show that the quality of the parameter estimation provided by ClusTrelHMM tends to be better than that provided by the Incremental Baum-Welch algorithm, a fixed topology algorithm, while additionally providing automatic topology reconfiguration of the model.

The ClusTrelHMM algorithm could be applied outside the scope of anomaly detection in network data because Hidden Markov Models have a wide range of applications. For example, ClusTrelHMM could extend the work from Hervieu et al. of modeling trajectories in video images [HBC07] and [HBC08] to model trajectories online.

In summary, ClusTrelHMM framework can be used to compute the parameters of HMMs incrementally without having to worry about the topology of the model.

5.2 Further work

This section describes two possible extensions to the work that was developed during this dissertation that were not explored due to time restrictions.

5.2.1 Exponential Forgetting

The online learning algorithms studied and developed in this work suffer from a particular symptom. The longer the algorithms maintain their normal operation, the slower the algorithms are to react to the changes in context of the observations that may occur.

The reason behind has to do with the sums present in the equations of IBW algorithm (2.24, 2.25 and 2.27) and present in the equations of the online QHMM (3.3 and 3.4). These sums grow with the number of observations and as such the values of the fractions of those equations change ever more slowly as time goes by.

A simple solution to this problem is to do exponential forgetting of the old contributions of those sums. In this way recent contributions would have a stronger impact in the estimation than the older ones.

Equation 5.1 demonstrates how the forgetting can be implemented. Instead of maintaining the normal sums of the previous equations it uses variables that are updated similarly to S_t. This variable is updated based on a value v_t which is equal to the last member of the previous sum and the variable at the previous moment in time S_{t-1}.

$$S_t = v_t + 2^{-\lambda} S_{t-1} \tag{5.1}$$

A comparative analysis of the algorithms with this exponential forgetting would help understand how much faster the implemented algorithms could react to the changes in concept of the observations.

5.2.2 Clustering Algorithms

In chapter 4, the reason for the application of ClusTrel was twofold. First, it served the purpose of computing the emission distributions. Second, it also allowed the exploration of more clustering possibilities due to its trellis structure.

It would be interesting to try Hervieu's approach for estimating the transition matrix and initial probability distributions without the associated restrictions of the trellis structure of ClusTrel.

For example, we could have looked only to the selected clustering structures of ClusTrel at each moment and adapt the variables of QHMM as it was done in 3.2. This approach would force the resizing of the parameters of the estimation procedure to be non trivial because it would have to include transition between clustering structures with undefined number of clusters. This would force the determination of which clusters are splitted, which cluster are merged and which maintain between each consecutives moments.

This possibility would have the advantage of being faster than ClusTrelHMM because there wouldn't be a need to compute 3 HMMs for each model with a certain number of states. But on the other hand, more time would have to be spent in finding which states were merged and which states were splitted so the structures could be resized accordingly.

Finally, the online clustering algorithm could even be abstracted to capitalize on future clustering algorithms and improvements.

Conclusion

Appendix A

Summary of Estimation Methods

This appendix presents a summary about recent methods for parameter estimation of Hidden Markov Models taken out from [KDSM09] but inserted here for reference (tables A.2 and A.2).

Name	Type	References	Pros	Cons	Comp. Cost
Particle MCMC	Bayesian	[ADH09]	Standard SMC applicable, only requires design of a proposal for θ.	Expensive.	$O(NT)$ per MCMC step
Smooth Likelihood Evaluation	MLE	[Pit02]	Simple to implement. Can use standard optimization packages.	Requires $n_x = 1$.	$O(N \log NT)$ per evaluation
Gradient	MLE	[PARM09] [PaAD06] [PDS05]	Generally applicable. Standard SMC smoothing algorithms applicable.	Difficult to tune scaling (especially if n_θ large). Locally optimal.	$O(NT)$ or $O(N^2T)$ per parameter update
EM	MLE	[BDM06] [ADST04] [OCDM06] [WSN08]	Standard SMC smoothing algorithms applicable.	Restricted model class. Locally optimal.	$O(NT)$ or $O(N^2T)$ per parameter update
Iterated Filtering	MLE	[IaAK09]	Standard SMC applicable.	Difficult to tune scaling (especially if n_θ large). Locally optimal.	$O(NT)$ per iteration

Table A.1: Table with a recent summary of the offline estimation methods [KDSM09].

Name	Type	References	Pros	Cons	Comp. Cost
Artificial Dynamics	Bayesian	[Hig85]	Standard SMC applicable	Target distribution altered. Difficult to tune introduced dynamics.	$O(NT)$
Resample-Move	Bayesian	[Sto02] [Fea02] [KBF+06]	Elegant. Target distribution unalterabled.	Restricted model class. Degeneracy problem. Informative priors n_θ small.	$O(NT)$
Gradient	MLE	[PDS05]	Asymptotically efficient. Generally applicable.	Expensive. Difficult to tune scaling (especially if n_θ large). Locally optimal.	$O(N^2)$ per parameter update
EM	MLE		Asymptotically efficient.	Restricted model class. Expensive. Locally optimal.	$O(N^2)$ per parameter update
EM pseudo	pseudo MLE	[ADT05]	Minimal tuning. No degeneracy for small L.	Requires stationary distribution. Loss of assymptotic efficiency.	$O(NL)$ per parameter update

Table A.2: Table with a recent summary of the online estimation methods [KDSM09].

References

[ADH09] C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov Chain Monte Carlo. *Genetics*, 72(May 2007):1–32, 2009.

[ADST04] C. Andrieu, A. Doucet, S. Singh, and V. Tadić. *Particle Methods for Change Detection, Identification and Control*, volume 92, pages 423–438. 2004.

[ADT05] C. Andrieu, A. Doucet, and V. Tadic. On-Line Parameter Estimation in General State-Space Models. *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 332–337, 2005.

[BDM06] M. Briers, A. Doucet, and S. Maskell. Smoothing algorithms for state-space models. *Ann. Inst. Stat. Math.*, pages 3062–3067, 2006.

[BE67] L. Baum and J. Eagon. An inequality with applications to statistical prediction for functions of Markov processes and to a model for ecology. *Bulletin of the American Mathematical Society*, 73:360–363, 1967.

[Bil98] J. Bilmes. A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. *International Computer Science Institute*, 4(510):126, 1998.

[Bis06] C. Bishop. *Pattern Recognition and Machine Learning*, volume 4. Springer, 2006.

[BOP97] M. Brand, N. Oliver, and A. Pentland. Coupled hidden Markov models for complex action recognition. *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 0:994–999, 1997.

[BPSW70] L. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 1970.

[Bra99] M. Brand. Structure Learning in Conditional Probability Models via an Entropic Prior and Parameter Extinction. *Neural Computation*, 11(5):1155–1182, 1999.

[Cap11] O. Cappé. Online EM Algorithm for Hidden Markov Models. *Journal of Computational and Graphical Statistics*, 20(3):728–749, 2011.

[Dig99] V. Digalakis. Online adaptation of hidden Markov models using incremental estimation algorithms. *IEEE Transactions On Speech And Audio Processing*, 7(3):253–261, 1999.

[Fea02] P. Fearnhead. MCMC, sufficient statistics and particle filters. *Journal Of Computational And Graphical Statistics*, 11(4):848–862, 2002.

[FL05] G. Florez-Larrahondo. *Incremental Learning of Discrete Hidden Markov Models*. PhD thesis, Mississippi State University, 2005.

[FM98] J. Ford and J. Moore. Adaptive estimation of HMM transition probabilities. *IEEE Transactions on Signal Processing*, 46(5):1374–1385, 1998.

[FMM12] A. Fernandes, S. Malinowski, and R. Morla. Online topology free Gaussian HMM parameter estimation based on clustering. *1st PhD. Students Conference in Electrical and Computer Engineering*, (142), 2012.

[GHS98] Y. Gotoh, M. Hochberg, and H. Silverman. Efficient training algorithms for HMMs using incremental estimation. *IEEE Transactions On Speech And Audio Processing*, 6(6):539–548, 1998.

[GM97] F. Le Gland and L. Mevel. Recursive estimation in HMMs. *Proc. IEEE Conf. Decis.Control*, pages 3468–3473, 1997.

[HBC07] A. Hervieu, P. Bouthemy, and J.P. Le Cadre. A HMM-based method for recognizing dynamic video contents from trajectories. *ICIP*, pages 2–5, 2007.

[HBC08] A. Hervieu, P. Bouthemy, and J.P. Le Cadre. A Statistical Video Content Recognition Method Using Invariant Features on Object Trajectories. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(11):1533–1543, 2008.

[Hig85] T. Higuchi. Self-organizing Time Series Model. *Compute*, 1985.

[IaAK09] E. Ionides and A. Bhadra ands A. King. Iterated Filtering. pages 1–19, 2009.

[JR90] B. Juang and L. Rabiner. The segmental K-means algorithm for estimating parameters of hidden Markov models. *Electrical Engineering*, 38(9):1639–1641, 1990.

[KBF+06] M. Klaas, M. Briers, N. Freitas, A. Doucet, and S. Maskell. *Fast particle smoothing: If I had a million particles*, pages 25–29. 2006.

[KDSM09] N. Kantas, A. Doucet, S. Singh, and J. Maciejowski. An overview of Sequential Monte Carlo Methods for Parameter Estimation in General State-Space Models. *Engineering*, 2009.

[KM93] V. Krishnamurthy and J. Moore. On-Line Estimation of Hidden Markov Model Parameters Based on the Kullback-Leibler Information Measure. *IEEE Transactions on Signal Processing*, 41(8):2557–2573, 1993.

[MD08] G. Mongillo and S. Deneve. Online Learning with Hidden Markov Models. *Neural Computation*, 20(7):1706–1716, 2008.

[MM12] S. Malinowski and R. Morla. A single pass trellis-based algorithm for clustering evolving data streams. *International Conference on Data Warehousing and Knowledge Discovery (DAWAK)*, September 2012.

[MZ97] I. MacDonald and W. Zucchini. *Hidden Markov and Other Models for Discrete-valued Time Series*. Chapman and Hall, 1997.

[OCDM06] J. Olsson, O. Cappé, R. Douc, and E. Moulines. Sequential Monte Carlo smoothing with application to parameter estimation in non-linear state space models. *Bernoulli*, 14(1):155–179, 2006.

[PaAD06] G. Poyiadjis and S. Singh ands A. Doucet. Gradient-free Maximum Likelihood Parameter Estimation with Particle Filters. *American Control Conference*, pages 3062–3067, 2006.

[PARM09] C. Pierre-Arnaud, D. Romain, and R. Munos. Sensitivity analysis in HMMs with application to likelihood maximization. *Advances in Neural Information Processing Systems*, pages 1–9, 2009.

[PBM02] A. Panuccio, M. Bicego, and V. Murino. A Hidden Markov Model-based approach to sequential data clustering. *Structural Syntactic and Statistical Pattern Recognition*, 2396:734–743, 2002.

[PDS05] G. Poyadjis, A. Doucet, and S. Singh. Maximum likelihood parameter estimation using particle. *In Proc. Joint Statistical Meeting*, 2005.

[Pit02] M. Pitt. Smooth Particle Filters for Likelihood Evaluation and Maximisation. *Warwick Economic research papers*, (651):1–43, 2002.

[Rab89] L. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[Ryd97] T. Rydén. On recursive estimation for Hidden Markov Models. *Stochastic Processes and their Applications*, 66(1):79–96, 1997.

[Sch02] A. Schliep. *A Bayesian Approach to Learning Hidden Markov Model Topology with Applications to Biological Sequence Analysis*. 2002.

[SRP+01] B. Stenger, V. Ramesh, N. Paragios, F. Coetzee, and J. Buhmann. Topology free hidden Markov models: application to background modeling. *Proceedings Eighth IEEE International Conference on Computer Vision ICCV 2001*, 00(C):294–301, 2001.

[Sto02] G. Storvik. Particle Filters for State-Space Models With the Presence of Unknown Static Parameters, 2002.

[VEJ97] Jr.R. Vasko and A. El-Jaroudi. *Hidden Markov Model Topology Inference: The DISSOLVE algorithm*. PhD thesis, Pittsburgh, PA, USA, 1997.

[Wel02] B. Wellman. Designing the Internet for a networked society. *Communications of the ACM*, 45(5):91–96, 2002.

[WSN08] A. Wills, T. Schon, and B. Ninness. Parameter Estimation for Discrete-Time Nonlinear Systems Using EM. *In Proc. 17th IFAC World Congress*, 2008.

REFERENCES