Université de Technologie de Compiègne

THÈSE

présentée en vue de l'obtention du Grade de

Docteur de l'Université de Technologie de Compiègne
Spécialité: Technologies de l'Information et des Systèmes

par

Faicel Chamroukhi

Hidden process regression for curve modeling, classification and tracking

Soutenue publiquement le 13 décembre 2010 devant le jury composé de :

Fabrice ROSSI	Professeur, TELECOM ParisTech	(Rapporteur)
Christophe BIERNACKI	Professeur, Université Lille 1	(Rapporteur)
Yves GRANDVALET	Directeur de recherche, CNRS	(Examinateur)
Gérard GOVAERT	Professeur, UTC	(Directeur de thèse)
Patrice AKNIN	Directeur de recherche, INRETS	(Directeur de thèse)
Allou SAMÉ	Chargé de recherche, INRETS	(Examinateur)

Acknowledgements

I would like to thank my advisors Mr. Gérard Govaert and Mr. Patrice Aknin for their guidance.

A very special thanks go out to my co-advisor Mr. Allou Samé for his support and for his availability.

I would also like to thank Mr. Christophe Biernacki and Mr. Fabrice Rossi for reviewing my thesis and for their valuable feedback. I am also grateful to Mr. Yves Grandvalet for agreeing to be part of my committee.

Special thanks to my current and former colleagues at LTN-INRETS laboratory. I extend my thanks to Mr. Marc Antoni from the SNCF infrastructure department for his collaboration with us.

I have been a teacher assistant at Université Paris 13 during my three years of thesis, I take this opportunity to thank my tutor Mme. Christine Choppy and all the professors I have worked with.

Finally, I express my thanks to my friends and my family.

Faicel Chamroukhi
Paris, December 10, 2010

Abstract

This research addresses the problem of diagnosis and monitoring for predictive maintenance of the railway infrastructure. In particular, the switch mechanism is a vital organ because its operating state directly impacts the overall safety of the railway system and its proper functioning is required for the full availability of the transportation system; monitoring it is a key task within maintenance team actions. To monitor and diagnose the switch mechanism, the main available data are curves of electric power acquired during several switch operations.

This study therefore focuses on modeling curve-valued or functional data presenting regime changes. In this thesis we propose new probabilistic generative machine learning methodologies for curve modeling, classification, clustering and tracking. First, the models we propose for a single curve or independent sets of curves are based on specific regression models incorporating a flexible hidden process. They are able to capture non-stationary (dynamic) behavior within the curves and address the problem of missing information regarding the underlying regimes, and the problem of complex shaped classes. We then propose dynamic models for learning from curve sequences to make decision and prediction over time. The developed approaches rely on autoregressive dynamic models governed by hidden processes. The learning of the models is performed in both a batch mode (in which the curves are stored in advance) and an online mode as the learning proceeds (in which the curves are analyzed one at a time). The obtained results on both simulated curves and the real-world switch operation curves demonstrate the practical use of the ideas introduced in this thesis.

Keywords: Curve modeling, regression, classification, clustering, finite mixture models, Hidden Markov Models, dynamical modeling, online learning, EM algorithms, diagnosis.

Contents

Notations

We list here the general notation used in the thesis. Deviations from this notation will be mentioned prior to use. In general, a vector is represented in upright bold (e.g, $\mathbf{x}, \mathbf{y}, \mathbf{z}, \ldots$). A vector of zeros of arbitrary length is denoted as $\mathbf{0}$, while a vector of ones is denoted as $\mathbf{1}$. All vectors are assumed to be column vectors. A transpose of a vector $\mathbf{x}$ is denoted by $\mathbf{x}^T$ so that $\mathbf{x}^T$ represents a row-vector. Matrices are represented in capitalized upright bold (e.g., $\mathbf{X}, \mathbf{Y}, \ldots$). A transpose of a matrix $\mathbf{A}$ is denoted by $\mathbf{A}^T$. The identity matrix of size n is denoted as $\mathbf{I}_n$. Scalar parameters are denoted by Greek letters (e.g., $\pi, \alpha, \beta, \tau, \ldots$) while vector parameters and sets of model parameters are denoted by boldface Greek letters (e.g., $\boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{\tau}, \boldsymbol{\Psi}, \boldsymbol{\Theta} \ldots$). Sets are denoted by calligraphic capital letters (e.g., $\mathcal{X}, \mathcal{Y}, \mathcal{Z}, \ldots$) except for when the sets already have established symbols (e.g., $\mathbb{R}$). Other notations used in this thesis are listed below:

General notations:

$\mathbb{E}[X]$	expected value of X
$\mathbb{E}[Y\|X = x]$	expected value of Y conditionally on $X = x$
$\mathcal{N}(\,.\,; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	multivariate Gaussian density with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$.
$\mathcal{M}(\,.\,; \boldsymbol{\pi})$	multinomial distribution with parameter vector $\boldsymbol{\pi}$
$L(\boldsymbol{\Psi}; \mathbf{X})$	likelihood function of the parameter vector $\boldsymbol{\Psi}$ for the data $\mathbf{X}$
$\mathcal{L}(\boldsymbol{\Psi}; \mathbf{X})$	log-likelihood function of the parameter vector $\boldsymbol{\Psi}$ for the data $\mathbf{X}$
$\hat{\boldsymbol{\Psi}}$	estimate of $\boldsymbol{\Psi}$
$\mathbf{A}_{ij}$	the jth column of the ith row of $\mathbf{A}$
$\mathbf{A}^{-1}$	inverse of $\mathbf{A}$
$\mathrm{trace}(\mathbf{A})$	trace of $\mathbf{A}$
$\det(\mathbf{A})$	determinant of $\mathbf{A}$

Multidimensional data:

$\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$	a sample of n observations
$\mathbf{x}_i$	ith observation
$\mathbf{z} = (z_1, \ldots, z_n)$	hidden class vector
$z_i = k \in \{1, \ldots, K\}$	class label of $\mathbf{x}_i$
K	number of hidden classes
$\mathbf{c} = (c_1, \ldots, c_n)$	class vector
G	total number of classes

Multidimensional sequential data:

$\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$	a observation sequence
$\mathbf{y}_t$	observation at time t
$\mathbf{z} = (z_1, \ldots, z_n)$	class (state) vector
$z_t = k \in \{1, \ldots, K\}$	class label of $\mathbf{y}_t$
K	number of classes (states)

Single curve:

$\mathbf{y} = (y_1, \ldots, y_m)$	a curve of m observations
y_j $(j = 1, \ldots, m)$	jth observation (point) of the curve
m	total number of observations for a single curve
$\mathbf{z} = (z_1, \ldots, z_m)$	an underlying hidden process for a single curve
$z_j = k \in \{1, \ldots, K\}$	class label of y_j $(j = 1, \ldots, m)$ (kth polynomial model (regime))
K	total number of regimes (segments)

Set of n curves:

$\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$	A Given training set of n curves
$\mathbf{y}_i = (y_{i1}, \ldots, y_{im})$	ith curve
y_{ij}	jth observation (point) of the ith curve
n	total number of curves
$\mathbf{c} = (c_1, \ldots, c_n)$	vector of classes (states, groups) for the n curves
$c_i = g \in \{1, \ldots, G\}$	class label of the ith curve
G	total number of classes (states, groups)
$\mathbf{h} = (h_1, \ldots, h_n)$	vector of sub-classes for a set of n curves issued from the same class
$h_i = r \in \{1, \ldots, R\}$	sub-class label of the ith curve
R	total number of sub-classes

Since online learning concerns the case when the data are arriving one at a time, we will denote by $\mathbf{y}_t$ the observed vector at time t $(t = 1, \ldots, T)$.

Probability distribution:

$p(\,.\,)$	generic notation of a probability density function (p.d.f)
$f(\,.\,)$	p.d.f
$f(\,.\,;\boldsymbol{\theta})$	parametric p.d.f

Chapter 1

Introduction

Contents

1.1 Context of the study

The results of this study were applied to the diagnosis of railway infrastructure. The large amount of infrastructure components, the spatial structure of the components, the variable environmental contexts, the multiple types of failure and, particularly, the strong financial constraints have led railway infrastructure managers to deploy monitoring systems *en masse* during recent decades. All of the infrastructure components are potentially concerned, including the rail, the track ballast, the track geometry, the switch, the catenary, and the signaling system. This research will address the particular problem of monitoring the switch mechanism.

The switch mechanism is a sensitive organ in railway infrastructure that enables (high-speed) trains to be guided from one track to another at a railway junction. Its proper functioning is required for the full availability of the transportation system. Moreover, it is a vital organ because its operating state directly impacts the overall safety of the railway system; monitoring it is a key task within maintenance team actions.

To monitor and diagnose the switch mechanism, condition measurements are acquired during several switch operations. The main available measurement is the electric power consumed during a switch operation, which is hereafter referred to as the "switch operation curve" (see Figure 1.1). Each switch operation consists of successive phases that are reflected in the curve shape as changes in regime (see Figure 1.1). The suc-

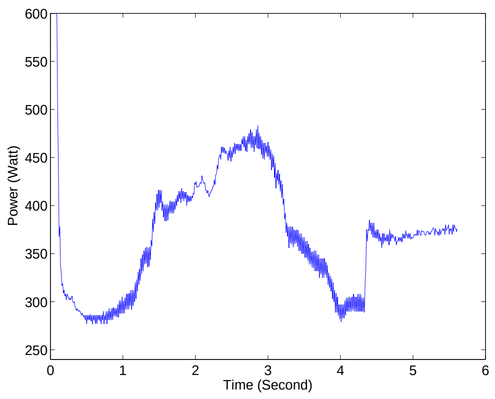


Figure 1.1: A curve showing electrical power consumed during a switch operation.

cession of phases in the switch operation curve constitutes the first temporal aspect of the data. Details on both the switch mechanism and the phases involved in a switch operation are given in Chapter 6, which is dedicated to this application.

The diagnosis task is achieved through the analysis of switch operation curves. In this framework, curve modeling and classification remain the key materials. In a perspective of switch monitoring, sequences of curves are analyzed. This results in a second temporal aspect with respect to the data that is concerned with the evolution through time of the system. Figure 1.2 illustrates this second temporal aspect.

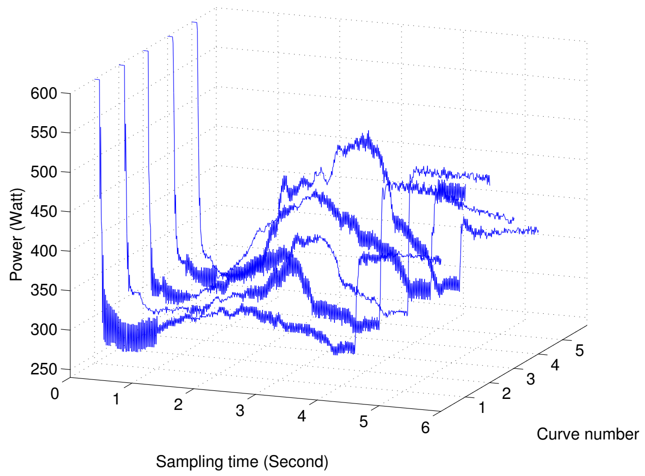


Figure 1.2: Examples of curves acquired during successive switch operations: the second temporal aspect.

Several approaches are possible for achieving the diagnosis task. The two main approaches are the model-based approach and the pattern recognition approach, which is based on machine learning. In the former, a physical model is built to describe the studied system, for example, by relying on automata theory (Isermann, 1984, 2004), and failure is detected by computing the difference (or residual) between the acquired measurement and the one simulated according to a pre-established physical model of the proper functioning state of the system. A binary decision is often assumed.

The latter approach, however, relies on pattern recognition to identify faults (Dubuisson, 1990). Very often, this pattern recognition task involves a learning problem in which we learn a decision rule, or a functional mapping between possibly labeled measurements and the considered operating states. This approach is particularly appropriate for the problem of multi-state diagnosis. In addition, it can easily address problems related to missing information and allows for the integration of prior knowledge, such as experts information. For these reasons, we adopt the second approach in this thesis.

The focus of this thesis is therefore on models for automatically extracting unknown, useful information (e.g., features, simplified models, and classes) from these curves to build reliable decision systems for predictive maintenance of the switches.

1.2 A machine learning context

The paradigm for learning from raw data is known as *machine learning* (Mitchell, 1997; Vapnik, 1999). Machine learning approaches for the acquisition of knowledge from data can be used for analysis, interpretation and prediction. In particular, statistical machine learning, pioneered by Vapnik and Chervonenkis (1974), is the field of machine learning distinguished by the fact that the data are assumed to be realizations of random variables so that the resulting models are statistical (probabilistic) models. It represents an elegant framework upon which high-performance information processing systems can be built. Indeed, the computer-aided decision systems that are built using this paradigm have shown their superior performance and reliability in many application domains, including signals (Bach and Jordan, 2006), images (Benboudjema and Pieczynski, 2007; Caillol et al., 1997; Kivinen et al., 2007; Smyth et al., 1994) and text (Aseervatham and Bennani, 2009; Chemudugunta et al., 2006; Joachims, 2002) as well as for large-scale data sets (Bordes, 2010).

To develop reliable decisions for the studied system and to make accurate decisions and predictions for future data, there is an important need to understand the processes underlying the curves. This therefore leads us to a statistical machine learning paradigm (Mitchell, 1997; Vapnik, 1999), particularly *generative* approaches (Jebara, 2001, 2003) such as regression models, mixture models and Hidden Markov Models, which form the core of this research.

For the diagnosis task, we are first interested in modeling the curves that include unknown regime changes. Therefore, as in addressing statistical machine learning in general applications, we are confronted with the problem of missing information. In the context of curve modeling, the missing information problem relates to the unobserved process governing the curve. This process is related to regime changes within a curve or a set of curves and can be treated by a *modeling/segmentation* approach. Therefore, at this stage, the term “dynamical” is used to indicate regime changes within the curves.

The second objective is to decide on the operating state of the new curves and detect possible faults. This is the curve *classification* problem, which is performed by assigning new curves to predefined operating states (or classes). The class labels of the data are often missing in real applications, which results in the curve *clustering* problem.

Furthermore, in this thesis, the curve classification task is addressed from two points of view. The classification rules for the first point of view are taken from *static* modeling techniques because the curves are assumed to be independent; however, the second one relies on building decision rules in a *dynamical* framework from curves acquired sequentially during the operating process. Hence, at this stage of modeling curves sequences, the term "dynamical" relates to the unobserved process representing state variation over time. Figure 1.2 illustrates state variation from one curve to another.

In these frameworks, which are distinguished by the unknown underlying regimes within the curves and by the operating states of the curves which are often missing, the built models are associated with the *latent data models* introduced by Spearman (1904), including mixture models (Frühwirth-Schnatter, 2006; McLachlan and Peel., 2000; Titterington et al., 1985) and Hidden Markov Models (Juang and Rabiner, 1985; Juang et al., 1985, 1986; Rabiner, 1989). The optimization of such models is performed in a maximum likelihood estimation framework. The maximum likelihood estimator is of major importance in statistical theory due to its attractive asymptotic properties for parameter estimation problems that it is asymptotically unbiased, normal and consistent. Particularly, for the latent data models, maximum likelihood estimation cannot often be performed analytically in a closed form. The Expectation-Maximization (EM) algorithms (Baum et al., 1970; Celeux and Govaert, 1992; Dempster et al., 1977; McLachlan and Krishnan, 1997; Neal and Hinton, 1998) provide an elegant tool for optimizing such models in both a batch mode and an online mode (Cappé and Moulines, 2009; Titterington, 1984).

1.3 Contributions and outline of the thesis

The chapters of this thesis are arranged as follows. Chapter 2 gives an overview of classification and clustering methods for multidimensional data from the existing literature, with a particular focus on the generative learning approaches, which are the basis of this thesis. We describe a majority of well-known models and practical techniques and show how they work. Sometimes, we provide further explanation(s) or other points of view, in particular from a probabilistic prospective (e.g., in the case of the piecewise regression model).

We then propose new approaches for curve modeling, classification and tracking. The main contributions of this thesis can be summarized in the responses for the following questions on both open methodological issues and a real-world application:

- How can we build adapted model(s) for representing and summarizing curves with regime changes?
- How can we define an accurate discrimination rule by considering both homogeneous and dispersed curves?

- When expert information is missing, how can we automatically search for possible classes from curves?
- How can we model the underlying dynamical behavior from sequential curves and then make decisions and predictions?

More specifically, we first propose in Chapter 3 a new approach for curve modeling that is based on a specific regression model that incorporates a discrete hidden logistic process (RHLP) (Chamroukhi et al., 2009c). The RHLP model tackles the dimensionality problem and simultaneously provides a simplified representation for each curve. In particular, the flexibility of the proposed dynamical regression model allows us to deal with both the problem of the quality of regime transition (i.e., smooth and/or abrupt) and the temporal location of the regime changes, with each regime being associated with a regression model. We then show how the available multidimensional data classification machinery, in particular Mixture models, can then be used to perform curve clustering via mixture model-based clustering (Banfield and Raftery, 1993; Fraley and Raftery, 2002; McLachlan and Basford, 1988) and curve classification via Mixture Discriminant Analysis (MDA) (Hastie and Tibshirani, 1996). The classification in these two contexts is performed in the space of curve descriptors. The proposed RHLP modeling approach is then compared to the existing methods for the subject, including the piecewise regression and the Hidden Markov Model Regression (HMMR). This model reveals superior performance in terms of curve modeling, segmentation and classification as compared to alternative approaches on simulated curves and in application to switch operation curves.

In Chapter 4, we further extend the RHLP model presented in Chapter 3 for a single curve to the case of a set of curves (Chamroukhi et al., 2010). In the resulting approach, there is no longer need for feature extraction because the curves are directly classified in the “functional” space (i.e., the space of curves) rather than the space of descriptors, as in the previously presented two-strategy approach. We therefore show how the curve classification can be performed through Functional Linear Discriminant Analysis (FLDA). From a practical point of view, this model clearly outperforms the classical regression models, including piecewise regression and spline regression. For the piecewise regression model, the use of dynamic programming may require an expensive computational load, and the spline regression is not dedicated to regime change detection. This model was assessed using a simulation study and an application on the real-world curves of switch operations.

Furthermore, we provide another extension of the RHLP model for learning from a set of heterogeneous curves by integrating it into a mixture formulation, which leads us to a mixture of regression models with a hidden logistic process, abbreviated as MixRHLP (see section 4.3). The MixRHLP model is adapted for capturing curve heterogeneity; it is useful for curve clustering and curve classification, especially in the case of complex shaped classes. The classification scheme in this case lies in Functional Mixture Discriminant Analysis (FMDA). The performance of the MixRHLP model in terms of curve clustering is evaluated by performing comparisons with existing methods, including polynomial regression mixture, polynomial spline regression mixture and piecewise polynomial regression mixture using simulated curves with regime changes.

Whereas the previous chapters are concerned with a *static* framework insofar as the curves are assumed to be independent, proposed *dynamical* models are built from a curve sequence in Chapter 5. For this dynamical modeling of curve sequences, we propose two models with respect to the type of underlying dynamical process under consideration. The first assumes a hidden logistic process, resulting in the Autoregressive RHLP (ARHLP) model. The second uses a non-homogeneous hidden Markov process, which was used to develop the Autoregressive non-homogeneous HMM (AR-NH-HMM). The data used by these models are the features extracted from each curve by the RHLP model presented in Chapter 3. We show how to make decisions and predictions with these models, and we provide a formulation of dedicated EM algorithms for learning model parameters. The learning task for these dynamical models is performed in both a batch mode (in which the curves are stored in advance) and an online mode (in which the curves are analyzed one at a time). At this stage, the model structure is fixed. The problem of building a model with an evolutionary structure is not addressed in this work.

The insights gained from these approaches are most relevant to the practical real-world diagnosis application presented in Chapter 6.

Chapter 2

State of the art

Contents

2.1 Introduction

In this chapter, we provide an overview of methods used to address data modeling and classification. The classification problem is considered in a supervised context (i.e., discrimination) and in an unsupervised context (i.e., clustering). The data may be multidimensional (i.e., independent observations or observation sequences) or functional (i.e., curves).

Two main approaches are generally used in the statistical learning literature for learning from rough data. They are known as the *discriminative* approach and the *generative* approach (Jebara, 2001, 2003). In classification, discriminative approaches learn a direct map from the inputs $\mathbf{x}$ to the output y, or they directly learn a model of the conditional distribution $p(y|\mathbf{x})$. From this conditional distribution, we can make predictions of y for any new value of $\mathbf{x}$ by using the Maximum A Posteriori (MAP) classification rule

$$\hat{y} = \arg\max_{y\in\mathcal{Y}} p(y|\mathbf{x}). \tag{2.1}$$

Generative classifiers learn a model of the joint distribution $p(\mathbf{x}, y)$ that consists of modeling the class conditional densities $p(\mathbf{x}|y)$ together with the prior probabilities $p(y)$. The required posterior class probabilities are then computed using Bayes' theorem

$$p(y|\mathbf{x}) = \frac{p(y)p(\mathbf{x}|y)}{\sum_{y'} p(y')p(\mathbf{x}|y')}. \tag{2.2}$$

Because the outputs y are not always available (i.e., they may be missing or hidden), generative approaches are therefore more suitable for unsupervised learning.

In this thesis, we focus on probabilistic models for data modeling, clustering and discrimination. In this chapter, we therefore describe some classical probabilistic models for data modeling, classification and clustering, including regression models, mixture models and Hidden Markov Models, which form the core of this thesis.

Most of the approaches in this chapter are well-known in the literature and are included to provide context for later chapters. This chapter is organized into three main parts. The first is concerned with multidimensional independent data modeling, classification and clustering. The second and last parts address the same problems for the case of multidimensional observation sequences and curves (functional data), respectively. In the last part, we provide a new extension of the piecewise regression model to a mixture model-based curve clustering framework.

2.2 Multidimensional data modeling using finite mixture models

2.2.1 Finite mixture models

Finite mixture models (McLachlan and Peel., 2000; Titterington et al., 1985) are an example of latent variable models, widely used in probabilistic modeling, machine learning and pattern recognition. They are very useful to model heterogeneous classes since they assume that each class is composed of sub-classes.

Model definition

Let z represent a discrete random variable (binomial or multinomial) which takes its values in the finite set $\mathcal{Z} = \{1, \ldots, K\}$. The finite mixture model decomposes the density of $\mathbf{x}$ into a weighted linear combination of K component densities. In a general setting, the mixture density of $\mathbf{x}$ is

$$\begin{aligned} f(\mathbf{x}) &= \sum_{k=1}^{K} p(z=k)p(\mathbf{x}|z=k) \\ &= \sum_{k=1}^{K} \pi_k f_k(\mathbf{x}), \end{aligned} \tag{2.3}$$

where $\pi_k = p(z=k)$ represents the probability that a randomly chosen data point was generated by component k. These quantities are nonnegative $\pi_k \geq 0 \ \forall k$, and are constrained to sum to one: $\sum_{k=1}^{K} \pi_k = 1$. The parameters $\pi_1, \ldots, \pi_K$ are referred to as *mixing proportions* and $f_1, \ldots, f_K$ are referred to as *component densities*. Each of the K component densities typically consists of a relatively simple parametric model $p(\mathbf{x}|z=k; \boldsymbol{\Psi}_k)$ (such as a Gaussian distribution with parameters $\boldsymbol{\Psi}_k = (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$). Figure 2.1 gives the graphical model representation for the mixture model.

From the mixture model formulation, it can therefore be seen that the mixture model allows for placing K component densities in the input space to approximate the true density. The location (mean) and shape (covariance matrix) of each of the K components can be fixed independently of each other. Therefore, mixtures provide

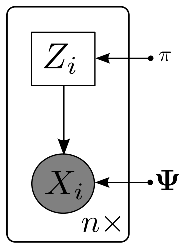


Figure 2.1: Graphical representation of a mixture model.

a natural generalization of the simple parametric density model which is global, to a weighted sum of these models, allowing local adaptation to the density of the data in the input space.

Parameter estimation

The common parameter estimation methods for mixture models are the maximum likelihood (McLachlan and Peel, 2000) and the Bayesian methods (Maximum A Posteriori (MAP)) where a prior distribution is assumed for the model parameters (Stephens, 1997, 2000). In this thesis, we consider the maximum likelihood framework. The optimization algorithm for performing the maximum likelihood parameter estimation is the Expectation-Maximization (EM) algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997). In the next section we will discuss the use of the Expectation-Maximization (EM) algorithm for learning the parameters of mixture models. The objective is to maximize the likelihood, or equivalently, the log-likelihood as a function of the model parameters $\mathbf{\Psi} = (\pi_1, \ldots, \pi_K, \mathbf{\Psi}_k, \ldots, \mathbf{\Psi}_K)$, over the parameter space $\mathbf{\Omega}$.

Assume we have an i.i.d sample $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$. The observed-data log-likelihood is then given by:

$$\begin{aligned} \mathcal{L}(\mathbf{\Psi}; \mathbf{X}) &= \log \prod_{i=1}^{n} p(\mathbf{x}_i; \mathbf{\Psi}) \\ &= \sum_{i=1}^{n} \log \sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i; \mathbf{\Psi}_k). \end{aligned} \tag{2.4}$$

In this case, due to the logarithm of the sum, the log-likelihood to be maximized results in a nonlinear function and there is no way to maximize it in a closed form. However, it can be locally maximized using iterative procedures such as gradient ascent, a Newton Raphson procedure or the Expectation-Maximization (EM) algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997). We will focus on the EM algorithm which is widely used and particularly adapted for mixture models. The next section presents the EM algorithm for general parametric mixture models before applying it to Gaussian mixtures.

2.2.2 The EM algorithm

The Expectation-Maximization (EM) algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997) is a broadly applicable approach to the iterative computation of maximum likelihood estimates in the framework of latent data models. In particular, the EM algorithm simplifies considerably the problem of fitting finite mixture models by maximum likelihood.

The EM algorithm for finite mixtures

The EM algorithm is an iterative algorithm where each iteration consists of two steps, the Expectation step (E-step) and the Maximization step (M-step). Within this incomplete-data framework, we let $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ denote a set of n multidimensional observed data and we let $\mathbf{z} = (z_1, \ldots, z_n)$ denote the corresponding unobserved (missing) labels where the class label z_i is drawn from a discrete-valued variable z which takes its values in the finite set $\mathcal{Z} = \{1, \ldots, K\}$. The probability of a single complete data point for a parametric mixture model with parameters $\mathbf{\Psi}$ is

$$\begin{aligned} p(\mathbf{x}, z) &= p(\mathbf{x}, z; \mathbf{\Psi}) \\ &= p(z)p\big(\mathbf{x}|z; \mathbf{\Psi}_z\big) \\ &= \pi_z f_z\big(\mathbf{x}; \mathbf{\Psi}_z\big). \end{aligned} \tag{2.5}$$

The log-likelihood of $\mathbf{\Psi}$ for the complete-data $(\mathbf{X}, \mathbf{z}) = ((\mathbf{x}_1, z_1), \ldots, (\mathbf{x}_n, z_n))$, called the complete-data log-likelihood, is therefore given by:

$$\mathcal{L}_c(\mathbf{\Psi}; \mathbf{X}, \mathbf{z}) = \log \prod_{i=1}^{n} p(\mathbf{x}_i, z_i; \mathbf{\Psi}). \tag{2.6}$$

In addition, since z_i belongs to $\{1, \ldots, K\}$, we can therefore rewrite the complete-data log-likelihood as

$$\begin{aligned} \mathcal{L}_c(\mathbf{\Psi}; \mathbf{X}, \mathbf{z}) &= \sum_{i=1}^{n} \log \prod_{k=1}^{K} \Big[p(z_i = k)p(\mathbf{x}|z_i = k; \mathbf{\Psi}_k)\Big]^{z_{ik}} \\ &= \sum_{i=1}^{n} \sum_{k=1}^{K} z_{ik} \log \pi_k f_k\big(\mathbf{x}_i; \mathbf{\Psi}_k\big), \end{aligned} \tag{2.7}$$

where z_{ik} is an indicator binary-valued variable such that $z_{ik} = 1$ if $z_i = k$ (i.e, when $\mathbf{x}_i$ is generated by the kth component density) and $z_{ik} = 0$ otherwise.

It can be seen that this log-likelihood depends on the unobservable data $\mathbf{z}$. The key idea of the EM algorithm is to perform the E-step by replacing $\mathcal{L}_c(\mathbf{\Psi}; \mathbf{X}, \mathbf{z})$ by its expectation conditionally on the observed data $\mathbf{X}$ and the current estimation $\mathbf{\Psi}^{(q)}$ of $\mathbf{\Psi}$, q being the current iteration.

The EM algorithm starts with an initial parameter $\mathbf{\Psi}^{(0)}$ and iteratively alternates between the two following steps until convergence:

E-step (Expectation): This step consists of computing the expectation of the complete-data log-likelihood (2.7), given the observations $\mathbf{X}$[1] and the current value $\mathbf{\Psi}^{(q)}$ of the

[1]We note that in this chapter, the bold upper matrix $\mathbf{X}$, used in Equation(2.8) denotes the set of training data examples $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and not a random variable.

parameter $\boldsymbol{\Psi}$ (q being the current iteration). This conditional expectation is often referred to as the Q-function. It is given by:

$$\begin{aligned} Q(\boldsymbol{\Psi}, \boldsymbol{\Psi}^{(q)}) &= \mathbb{E}\left[\mathcal{L}_c(\boldsymbol{\Psi}; \mathbf{X}, \mathbf{z})|\mathbf{X}; \boldsymbol{\Psi}^{(q)}\right] \\ &= \sum_{i=1}^{n}\sum_{k=1}^{K} \mathbb{E}[z_{ik}|\mathbf{x}_i, \boldsymbol{\Psi}^{(q)}] \log \pi_k f_k\big(\mathbf{x}_i; \boldsymbol{\Psi}_k\big) \\ &= \sum_{i=1}^{n}\sum_{k=1}^{K} p(z_{ik} = 1|\mathbf{x}_i; \boldsymbol{\Psi}^{(q)}) \log \pi_k f_k\big(\mathbf{x}_i; \boldsymbol{\Psi}_k\big) \\ &= \sum_{i=1}^{n}\sum_{k=1}^{K} \tau_{ik}^{(q)} \log \pi_k f_k\big(\mathbf{x}_i; \boldsymbol{\Psi}_k\big) \\ &= \sum_{i=1}^{n}\sum_{k=1}^{K} \tau_{ik}^{(q)} \log \pi_k + \sum_{i=1}^{n}\sum_{k=1}^{K} \tau_{ik}^{(q)} \log f_k\big(\mathbf{x}_i; \boldsymbol{\Psi}_k\big), \end{aligned} \quad (2.8)$$

where

$$\tau_{ik}^{(q)} = p(z_i = k|\mathbf{x}_i; \boldsymbol{\Psi}^{(q)}) = \frac{\pi_k f_k\big(\mathbf{x}_i; \boldsymbol{\Psi}_k^{(q)}\big)}{\sum_{\ell=1}^{K} \pi_\ell f_\ell\big(\mathbf{x}_i; \boldsymbol{\Psi}_\ell^{(q)}\big)} \quad (2.9)$$

is the posterior probability that $\mathbf{x}_i$ originates from the kth component density. Note that, in computing $\mathbb{E}[z_{ik}|\mathbf{x}_i, \boldsymbol{\Psi}^{(q)}]$, we used the fact that conditional expectations and conditional probabilities are the same for the indicator binary-valued variables z_{ik}: $\mathbb{E}[z_{ik}|\mathbf{x}_i, \boldsymbol{\Psi}^{(q)}] = p(z_{ik} = 1|\mathbf{x}_i, \boldsymbol{\Psi}^{(q)})$. As shown in the expression of Q, this step simply requires the computation of the conditional posterior probabilities $\tau_{ik}^{(q)}$.

M-step (Maximization): The M-step updates the estimate of $\boldsymbol{\Psi}$ by the value $\boldsymbol{\Psi}^{(q+1)}$ of $\boldsymbol{\Psi}$ that maximizes the Q-function $Q(\boldsymbol{\Psi}, \boldsymbol{\Psi}^{(q)})$ with respect to $\boldsymbol{\Psi}$ over the parameter space $\boldsymbol{\Omega}$:

$$\boldsymbol{\Psi}^{(q+1)} = \arg\max_{\boldsymbol{\Psi}\in\boldsymbol{\Omega}} Q(\boldsymbol{\Psi}, \boldsymbol{\Psi}^{(q)}). \quad (2.10)$$

Letting $Q_\pi(\pi_1, \ldots, \pi_K, \boldsymbol{\Psi}^{(q)})$ denotes the term in Q that is a function of the mixing proportions $(\pi_1, \ldots, \pi_K)$ and $Q_{\boldsymbol{\Psi}_k}(\boldsymbol{\Psi}_k, \boldsymbol{\Psi}^{(q)})$ denote the term in Q that is a function of the kth component density $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, we obtain:

$$Q(\boldsymbol{\Psi}, \boldsymbol{\Psi}^{(q)}) = Q_\pi(\pi_1, \ldots, \pi_K, \boldsymbol{\Psi}^{(q)}) + \sum_{k=1}^{K} Q_{\boldsymbol{\Psi}_k}(\boldsymbol{\Psi}_k, \boldsymbol{\Psi}^{(q)}) \quad (2.11)$$

where

$$Q_\pi(\pi_1, \ldots, \pi_K, \boldsymbol{\Psi}^{(q)}) = \sum_{i=1}^{n}\sum_{k=1}^{K} \tau_{ik}^{(q)} \log \pi_k \quad (2.12)$$

and

$$Q_{\boldsymbol{\Psi}_k}(\boldsymbol{\Psi}_k, \boldsymbol{\Psi}^{(q)}) = \sum_{i=1}^{n} \tau_{ik}^{(q)} \log f_k\big(\mathbf{x}_i; \boldsymbol{\Psi}_k\big) \quad (2.13)$$

for $k = 1, \ldots, K$. Thus, the maximization of the function $Q(\mathbf{\Psi};\mathbf{\Psi}^{(q)})$ w.r.t $\mathbf{\Psi}$ can be performed by separately maximizing Q_π with respect to the mixing proportions $(\pi_1, \ldots, \pi_K)$ and $Q_{\mathbf{\Psi}_k}$ with respect to parameters $\mathbf{\Psi}_k$ for each of the K components densities. The function Q_π is maximized with respect to $(\pi_1, \ldots, \pi_K) \in [0,1]^K$ subject to the constraint $\sum_k \pi_k = 1$. This maximization is done in a closed form and leads to

$$\pi_k^{(q+1)} = \frac{\sum_{i=1}^n \tau_{ik}^{(q)}}{n} = \frac{n_k^{(q)}}{n}, \tag{2.14}$$

where $n_k^{(q)}$ can be viewed as the expected cardinal number of the subpopulation k estimated at iteration q.

Now, let us consider the update for the parameters $\mathbf{\Psi}_k$ for each component density k. In general, the E- and M-steps have particularly simple forms when the complete-data probability density function is from the exponential family (McLachlan and Krishnan, 1997) and the solution to the M-step exists in a closed form. In the cases where the M-step can not be performed directly, adapted extensions can be used. Some extensions associated with the E- and the M- steps will be given in section 2.2.3.

The next section presents the EM algorithm for Gaussian mixtures.

The EM algorithm for Gaussian mixtures

Let us consider the Gaussian mixture model case, that is,

$$f(\mathbf{x}_i;\mathbf{\Psi}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_k,\mathbf{\Sigma}_k), \tag{2.15}$$

Figure 2.2 shows the corresponding graphical model representation with $\pi = (\pi_1, \ldots, \pi_K)$, $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K)$ and $\mathbf{\Sigma} = (\mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_K)$, and Figure 2.3 shows as example of a three-component Gaussian mixture density in $\mathbb{R}^2$ with $\pi = (0.3, 0.4, 0.3)$, $\boldsymbol{\mu}_1 = (-1.5, 0.6)$, $\boldsymbol{\mu}_2 = (0.1, 0.1)$, $\boldsymbol{\mu}_3 = (2.2, 1.1)$, $\mathbf{\Sigma}_1 = (0.4\ 0; 0\ 0.2)$, and $\mathbf{\Sigma}_2 = \mathbf{\Sigma}_3 = 0.3 \times \mathbf{I}_2$.

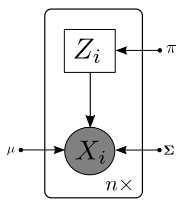


Figure 2.2: Graphical representation of a Gaussian mixture model.

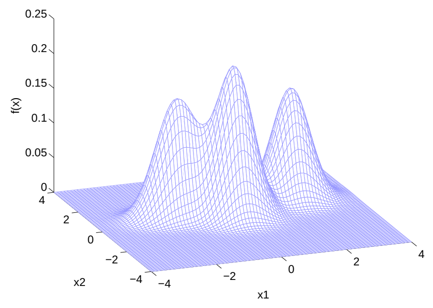


Figure 2.3: An example of a three-component Gaussian mixture density in $\mathbb{R}^2$.

The observed-data log-likelihood of $\mathbf{\Psi}$ for the Gaussian mixture model is given by

$$\mathcal{L}(\mathbf{\Psi};\mathbf{X}) = \sum_{i=1}^{n} \log \sum_{k=1}^{K} \pi_k \mathcal{N}\big(\mathbf{x}_i;\boldsymbol{\mu}_k,\mathbf{\Sigma}_k\big). \tag{2.16}$$

This log-likelihood is iteratively maximized by the EM algorithm. The complete-data log-likelihood in this case is given by:

$$\mathcal{L}_c(\mathbf{\Psi};\mathbf{X},\mathbf{z}) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik} \log \pi_k \mathcal{N}\big(\mathbf{x}_i;\boldsymbol{\mu}_k\mathbf{\Sigma}_k\big). \tag{2.17}$$

Thus, given an initial parameter $\mathbf{\Psi}^{(0)} = (\pi_1^{(0)},\ldots,\pi_K^{(0)},\mathbf{\Psi}_1^{(0)},\ldots,\mathbf{\Psi}_K^{(0)})$ where $\mathbf{\Psi}_k^{(0)} = (\boldsymbol{\mu}_k^{(0)},\mathbf{\Sigma}_k^{(0)})$, the EM algorithm alternates between the following steps until convergence to a local maximum of the log-likelihood function.

E-step: This step consists of computing the expectation of the complete-data log-likelihood (2.17), given the observations $\mathbf{X}$ and the current value $\mathbf{\Psi}^{(q)}$ of the parameter $\mathbf{\Psi}$:

$$\begin{aligned} Q(\mathbf{\Psi},\mathbf{\Psi}^{(q)}) &= \mathbb{E}\Big[\mathcal{L}_c(\mathbf{\Psi};\mathbf{X},\mathbf{z})|\mathbf{X};\mathbf{\Psi}^{(q)}\Big] \\ &= \sum_{i=1}^{n}\sum_{k=1}^{K}\tau_{ik}^{(q)}\log\pi_k + \sum_{i=1}^{n}\sum_{k=1}^{K}\tau_{ik}^{(q)}\log\mathcal{N}\left(\mathbf{x}_i;\boldsymbol{\mu}_k,\mathbf{\Sigma}_k\right). \end{aligned} \tag{2.18}$$

This step therefore computes the posterior probabilities

$$\tau_{ik}^{(q)} = p(z_i = k|\mathbf{x}_i,\mathbf{\Psi}^{(q)}) = \frac{\pi_k\mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_k^{(q)},\mathbf{\Sigma}_k^{(q)})}{\sum_{\ell=1}^{K}\pi_\ell\mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_\ell^{(q)},\mathbf{\Sigma}_\ell^{(q)})} \tag{2.19}$$

that $\mathbf{x}_i$ originates from the kth component density.

M-step: The M-step updates the estimate of $\mathbf{\Psi}$ by the value $\mathbf{\Psi}^{(q+1)}$ of $\mathbf{\Psi}$ that maximizes the function $Q(\mathbf{\Psi},\mathbf{\Psi}^{(q)})$ with respect to $\mathbf{\Psi}$ over the parameter space $\mathbf{\Omega}$. In this case of Gaussian mixture, the maximization of the Q-function with respect to $\boldsymbol{\mu}_k$ and $\mathbf{\Sigma}_k$ yields to the following updating formulas (McLachlan and Peel., 2000):

$$\boldsymbol{\mu}_k^{(q+1)} = \frac{1}{n_k^{(q)}}\sum_{i=1}^{n}\tau_{ik}^{(q)}\mathbf{x}_i, \tag{2.20}$$

$$\mathbf{\Sigma}_k^{(q+1)} = \frac{1}{n_k^{(q)}}\sum_{i=1}^{n}\tau_{ik}^{(q)}(\mathbf{x}_i-\boldsymbol{\mu}^{(q+1)})(\mathbf{x}_i-\boldsymbol{\mu}^{(q+1)})^T. \tag{2.21}$$

The E- and M-steps are alternated iteratively until the change in the log likelihood value are less than some specified threshold. The pseudo code 1 summarizes the EM algorithm for Gaussian mixture models.

Algorithm 1 Pseudo code of the EM algorithm for Gaussian mixture models.

Inputs: a data set $\mathbf{X}$ and the number of clusters K

```
1: fix a threshold $\epsilon > 0$
2: set $q \leftarrow 0$ (iteration)
3: Initialize: $\mathbf{\Psi}^{(0)} = (\pi_1^{(0)}, \ldots, \pi_K^{(0)}, \mathbf{\Psi}_1^{(0)}, \ldots, \mathbf{\Psi}_K^{(0)})$ with $\mathbf{\Psi}_k^{(0)} = (\boldsymbol{\mu}_K^{(0)}, \mathbf{\Sigma}_K^{(0)})$
4: while increment in log-likelihood $> \epsilon$ do
5:     E-step:
6:     for $k = 1, \ldots, K$ do
7:         Compute $\tau_{ik}^{(q)}$ for $i = 1, \ldots, n$ using Equation (2.19)
8:     end for
9:     M-step:
10:    for $k = 1, \ldots, K$ do
11:        Compute $\pi_k^{(q+1)}$ using Equation (2.14)
12:        Compute $\boldsymbol{\mu}_k^{(q+1)}$ using Equation (2.20)
13:        Compute $\mathbf{\Sigma}_k^{(q+1)}$ using Equation (2.21)
14:    end for
15:    $q \leftarrow q + 1$
16: end while
```

Outputs: $\hat{\mathbf{\Psi}} = \mathbf{\Psi}^{(q)}$

$\hat{\tau}_{ik} = \tau_{ik}^{(q)}$ (a fuzzy partition of the data)

Initialization Strategies and stopping rules

The initialization of the EM algorithm is a crucial point since it maximizes locally the log-likelihood. Therefore, if the initial value is inappropriately selected, the EM algorithm may lead to an unsatisfactory estimation. To address this issue, several methods are reported in the literature. The most used strategy consists of using several EM tries and selecting the solution maximizing the log-likelihood among those runs. For each run of EM, one can initialize it using for example one strategy among the following:

- Initialize randomly
- Compute a parameter estimate from another clustering algorithm such as K-means, Classification EM (Celeux and Diebolt, 1985), Stochastic EM (Celeux and Govaert, 1992)...
- Initialize EM with a few number of steps of EM itself.

Further details about choosing starting values for the EM algorithm for Gaussian mixtures can be found for example in Biernacki et al. (2003).

The EM algorithm can be stopped when the relative increase of the log-likelihood between two iterations is below a fixed threshold $|\frac{\mathcal{L}^{(q+1)} - \mathcal{L}^{(q)}}{\mathcal{L}^{(q)}}| \leq \epsilon$ or when a predefined number of iterations is reached.

The EM algorithm always monotonically increases the observed-data log-likelihood (Dempster et al., 1977; McLachlan and Krishnan, 1997). The sequence of parameter estimates generated by the EM algorithm converges toward at least a local maximum or a stationary value of the incomplete-data likelihood function (Wu, 1983). For the particular case of Gaussian mixtures, the EM algorithm converges toward a local maximum or a saddle point of the log-likelihood function (Xu and Jordan, 1996).

The EM algorithm has a number of advantages, including its numerical stability, simplicity of implementation and reliable convergence. In addition, by using adapted initializations, one may attempt the global optimum of the log-likelihood function. In general, both the E- and M-steps will have particularly simple forms when the complete-data probability density function is from the exponential family (McLachlan and Krishnan, 1997).

Some of the drawbacks of the EM algorithm consist of the fact that it is sometimes very slow to converge especially for high dimensional data; and in some problems, the E- or M-step may be analytically intractable. Fortunately, there exist extended EM versions that can tackle these problems. In the next section, we shall briefly address these issues by describing some EM extensions.

2.2.3 EM extensions

The EM variants mainly aim at increasing the convergence speed of EM and addressing the optimization problem in the M-step, or at computing the E-step when it is intractable. In the first case, one can speak about deterministic algorithms, e.g., Incremental EM (IEM) (Neal and Hinton, 1998), Gradient EM (Lange, 1995), Generalized EM (GEM) algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997), Expectation Conditional Maximization (ECM) (Meng and Rubin, 1993) and Expectation Conditional Maximization Either (ECME) (Liu and Rubin, 1994). In the second case, one can speak about stochastic algorithms, e.g., Monte Carlo EM (MCEM) (Wei and Tanner, 1990), Stochastic EM (SEM) (Celeux and Diebolt, 1985; Celeux et al., 1995, 1996) and Simulated Annealing EM (SAEM) (Celeux and Diebolt, 1991, 1992; Celeux et al., 1995, 1996). Therefore, the purpose of the deterministic variants of EM is on one hand to increase the speed of convergence of the basic EM, and on the other hand, to address the problem of the M-step. Indeed, for the E-step, the EM algorithm reveals to be impractical for application with large databases. Then the use of partial E-steps (e.g., Incremental) as suggested by Neal and Hinton (1998), in which the E-step is performed only for a block of data (or a single data point) at each iteration, can significantly reduce the computational time with preserving the convergence properties.

When the E-step is intractable, principal extended EM approaches use Monte-Carlo approximation, e.g., MCEM which approximates the analytically intractable E-step by the Monte-Carlo average, SEM which arises for the particular case of MCEM when one single simulation is drawn at each iteration and SAEM which results from a compromise between estimates provided by SEM and EM.

The M-step can be addressed by a Gradient EM which includes a gradient or a Newton Raphson procedure for the optimization of the M-step. The GEM algorithm consists of only improving the Q-function at the M-step rather than maximizing it at each EM iteration. Among the variety of approaches to accelerating EM that are concerned with the M-step, one can cite the ECM algorithm or the ECME algorithm. In ECM, the M-step is performed by several conditional maximizations (CM steps) by dividing the parameter space into sub-spaces. The parameter vector updates are then performed sequentially, one coordinate block after another in each sub space. ECME further extends the ECM algorithm where each CM-step either maximizes the Q-function or the incomplete-data (observed-data) log-likelihood function.

2.2.4 Online versions of the EM algorithm

The EM algorithms described in the previous section require that either the E-step or the M-step be performed for all data items. This batch mode learning is unsuitable in real-time applications where the data do arrive sequentially, one at a time. In this framework, implementing online (recursive) algorithms is a requirement to give quick parameter updates and to handle real-time problems. In this section we will describe recursive implementations of the EM algorithm.

Stochastic Gradient EM Algorithms

The first recursive parameter estimation procedure for incomplete data model has been proposed by Titterington (1984). Given n independent and identically distributed observations $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$, the recursive procedure is given by[1] :

$$\boldsymbol{\Psi}^{(t+1)} = \boldsymbol{\Psi}^{(t)} + \lambda_t [I_c(\boldsymbol{\Psi}^{(t)})]^{-1} \nabla_{\boldsymbol{\Psi}} \log p(\mathbf{x}_{t+1}; \boldsymbol{\Psi}^{(t)}) \tag{2.22}$$

where $\boldsymbol{\Psi}^{(t)}$ denotes the current value of the parameter estimate after t observations, the $\{\lambda_t\}$ is a decreasing sequence of positive step sizes and

$$I_c(\boldsymbol{\Psi}) = -\mathbb{E}\left[\nabla^2_{\boldsymbol{\Psi}} \log p(\mathbf{x}, z; \boldsymbol{\Psi})\right]$$

is Fisher information matrix associated with a complete observation $(\mathbf{x}, z)$.

The recursion (2.22) is recognizable as a stochastic approximation procedure on $\boldsymbol{\Psi}$ in the sense where the updating term only depends on the observation at time $t+1$. Indeed, it takes the typical form of the general stochastic algorithm which can be written as

$$\boldsymbol{\Psi}^{(t+1)} = \boldsymbol{\Psi}^{(t)} + \gamma_t \nabla_{\boldsymbol{\Psi}} J(\mathbf{x}_{t+1}; \boldsymbol{\Psi}^{(t)}), \tag{2.23}$$

where $\{\gamma_t\}$ is the learning rate of the algorithm (which corresponds to a matrix in (2.22)) and the criterion $J(\mathbf{x}_{t+1}; \boldsymbol{\Psi}^{(t)})$ measures the quality of the model described by the parameter $\boldsymbol{\Psi}^{(t)}$ given the observation $\mathbf{x}_{t+1}$ (Bottou, 1998, 2004). For further overview of the stochastic gradient algorithms the reader is referred to Bottou (1998, 2004).

[1] Here since in the online framework the data are arriving one at a time, we use the notation t to denote the number of the iteration processed with the data point $\mathbf{x}_t$.

Wang and Zhao (2006) have established that for mixture models of regular exponential family, under very mild conditions, the algorithm (2.22) converges almost surely toward a parameter vector from the set

$$\{\mathbf{\Psi} : \frac{\partial}{\partial \mathbf{\Psi}} \mathbb{E}[\log p(\mathbf{x}; \mathbf{\Psi})] = 0\}$$

which may contain local maxima and saddle points.

Online EM algorithm

A recursive parameter estimation using an online EM algorithm has been proposed by Sato and Ishii (2000) for normalized Gaussian networks. More recently, Cappé and Moulines (2009) proposed an online EM algorithm for latent data models, including mixtures. Cappé and Moulines (2009) proposed an extension of Titterington's approach (Titterington, 1984). It consists of replacing the expectation step by a stochastic approximation step, while keeping the maximization step unchanged. Formally, at iteration t, one computes the following function

$$Q_{t+1}(\mathbf{\Psi}, \mathbf{\Psi}^{(t)}) = Q_t(\mathbf{\Psi}, \mathbf{\Psi}^{(t-1)}) + \lambda_t\Big(\mathbb{E}\big[\log p(\mathbf{x}_{t+1}, z_{t+1}; \mathbf{\Psi})|\mathbf{x}_{t+1}; \mathbf{\Psi}^{(t)}\big] - Q_t(\mathbf{\Psi}, \mathbf{\Psi}^{(t-1)})\Big) \tag{2.24}$$

and computes the parameter $\mathbf{\Psi}^{(t+1)}$ by maximizing $Q_{t+1}(\mathbf{\Psi}, \mathbf{\Psi}^{(t)})$ w.r.t $\mathbf{\Psi}$. The advantage of (2.24) compared to (2.22) is that it automatically satisfies the parameter constraints (Cappé and Moulines, 2009). In addition, (2.24) does not explicitly require a matrix inversion compared to (2.22). Further practical comparisons, and comparisons in terms of rate of convergence between these two approaches are given in Cappé and Moulines (2009).

In the case where the complete-data likelihood belongs to the exponential family, Cappé and Moulines (2009) have established that the online EM algorithm arizing from (2.24) consists of updating the expectation of the complete-data sufficient statistics of the model parameters in the E-step by the following approximation recursion

$$\boldsymbol{S}_{t+1} = (1 - \lambda_t)\boldsymbol{S}_t + \lambda_t \mathbb{E}[\boldsymbol{T}(\mathbf{x}_{t+1}, z_{t+1}; \mathbf{\Psi})|\mathbf{x}_{t+1}, \mathbf{\Psi}^{(t)}] \tag{2.25}$$

and then run the M-step basing on the expected complete-data sufficient statistics. In (2.25), $\boldsymbol{S}_t$ represents of the expectation of the complete-data sufficient statistics computed up to time t and $\boldsymbol{T}(\mathbf{x}_{t+1}, z_{t+1}; \mathbf{\Psi})$ represents the sufficient statistic associated with the current complete observation $(\mathbf{x}_{t+1}, z_{t+1})$.

Cappé and Moulines (2009) proved that, if the complete data likelihood belongs to the exponential family and given suitable conditions on the stepsize λ_t, the online EM algorithm (2.25) is guaranteed to converge to a local optimum of the limiting normalised log-likelihood criterion (see (Cappé and Moulines, 2009)). However, the online EM does not guarantee increasing the incomplete-data log-likelihood after each update.

In the framework of mixture model-based clustering, Samé et al. (2007) proposed an online algorithm which consists of a stochastic gradient ascent derived from the Classification EM algorithm (CEM) and maximizes the expectation of the complete-data (classification) likelihood. For the case of Hidden Markov Models, recent online

EM implementations have been proposed for the case of continuous HMMs (Cappé, 2009) or discrete HMMs (Mongillo and Deneve, 2008).

2.3 Multidimensional data clustering

Clustering is often referred to as unsupervised learning in the sense that the class labels of the data are unknown (missing, hidden). Therefore, unsupervised learning algorithms, including clustering and segmentation are suitable for many applications where labeled data is difficult to obtain. Clustering is also often used to explore and characterize a dataset before running a supervised learning task. In typical clustering algorithms, the data are grouped by some notion of dissimilarity. Therefore, a similarity measure must be defined based on the data. The definition of dissimilarity and the method in which the data are clustered differ based on the clustering algorithm being applied.

Given a dataset of n i.i.d individuals (inputs) $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, the aim of clustering is to find a partition of the data by dividing them into clusters (groups) such that the data within a group tend to be more similar to one another as compared to the data belonging to different groups. In this section, we describe some well-known clustering algorithms, including distance-based and model-based clustering approaches.

2.3.1 K-means algorithm

The K-means algorithm (MacQueen, 1967), a straightforward and widely used clustering algorithm, is one of the most important algorithms in unsupervised learning. K-means is an iterative clustering algorithm that partitions a given dataset into a predefined number of clusters K. Typically, the value for K is chosen by prior knowledge of how many clusters actually appear in the data, how many clusters are desired for the current application, or the types of clusters found by experimenting with different values of K. In K-means, each cluster is represented by its mean (cluster centroid) $\boldsymbol{\mu}_k$ in $\mathbb{R}^d$. The default measure of similarity for K-means is the Euclidean distance $\|.\|^2$. It attempts to minimize the following nonnegative objective function referred to as *distortion measure*:

$$J(\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K, \mathbf{z}) = \sum_{k=1}^{K} \sum_{i=1}^{n} z_{ik} \|\mathbf{x}_i - \boldsymbol{\mu}_k\|^2 \tag{2.26}$$

which corresponds to the total squared Euclidean distance between each data point $\mathbf{x}_i$ and its closest cluster representative $\boldsymbol{\mu}_{z_i}$. After starting with an initial solution $(\boldsymbol{\mu}_1^{(0)}, \ldots, \boldsymbol{\mu}_K^{(0)})$ (eg, by randomly choosing K points in $\mathbb{R}^d$ or some data points), K-means iteratively performs the minimization of (2.26) by alternating between the two following steps until convergence:

- **Step 1:** Assignment step: Each data point is assigned to its closest centroid using the Euclidian distance: $\forall i = 1, \ldots, n$

$$z_{ik}^{(q)} = \begin{cases} 1 \text{ if } k = \arg\min_{z \in \mathcal{Z}} \|\mathbf{x}_i - \boldsymbol{\mu}_z\|^2 \\ 0 \text{ otherwise.} \end{cases} \tag{2.27}$$

- **Step 2:** Relocation step: Each cluster representative is relocated to the center (i.e., arithmetic mean) of all data points assigned to it:

$$\boldsymbol{\mu}_k^{(q+1)} = \frac{\sum_{i=1}^n z_{ik}^{(q)} \mathbf{x}_i}{\sum_{i=1}^n z_{ik}^{(q)}},$$

q being the current iteration.

The K-means algorithm is simple to implement and relatively fast. It converges when the assignments (and hence the centroids $\boldsymbol{\mu}$) no longer change. One can show that the K-means objective function (2.26) will decrease whenever there is a change in the assignment or the relocation steps. Convergence is guaranteed in a finite number of iterations since there are only a finite number of possible assignments for the set of discrete variables z_{ik}. However, the algorithm converges toward a local minimum because the minimized objective function is non-convex which implies that the algorithm is sensitive to the initial centroid locations. The convergence properties of K-means can be found in Bottou and Bengio (1995); Dang (1998). The local minima problem for K-means can be countered by running the algorithm multiple times with different initializations and then selecting the best result in the sense of the minimal distortion. Another limitation of K-means lies in the fact that, when running K-means, particularly with large values of K and/or the dimension of the input space is very high, one can have empty clusters. In these cases, reinitializing the cluster representative of the empty cluster or taking some points from the largest cluster, are possible.

From a probabilistic point of view, K-means is equivalent to a particular case of the CEM algorithm (Celeux and Govaert, 1992) (see section 2.3.2) for a mixture of K Gaussian densities with the same proportions $\pi_k = \frac{1}{K}\ \forall k$ and identical isotropic covariance matrices $\boldsymbol{\Sigma}_k = \sigma^2\mathbf{I}\ \forall k$. More precisely, if the soft assignments of data points to the mixture components are instead performed in a "hard" way so that each data point is assigned to the most likely mixture component (MAP principle), then one obtains the K-means algorithm. Therefore, K-means assumes that the dataset is composed of a mixture of K hyper-spheres of data where each of the K clusters corresponds to one of the mixture components. This implies that K-means will not provide good performance when the data can not be described by well separated spherical Gaussians.

Online *K*-means: The online (sequential) version of K-means is run as follows. When the data point $\mathbf{x}_t$ is presented at time t[1], the cluster center $\boldsymbol{\mu}_{k^*}$ ($k^* \in \{1, \ldots, K\}$) that is the nearest to $\mathbf{x}_t$ in the sense of the minimum distance (Euclidean) to the data

[1]Here since we are in an online framework, we use the index t instead of q to denote the iteration which in this case represents the index number of the data point.

point is then updated by the following rule:

$$\boldsymbol{\mu}_{k^*}^{(t+1)} = \frac{\mathbf{x}_t + (\#\text{class } k^*)^{(t)} \boldsymbol{\mu}_{k^*}^{(t)}}{(\#\text{class } k^*)^{(t)} + 1} \tag{2.28}$$

where $(\#\text{class } k^*)^{(t)}$ is the number of data points assigned to the cluster k at time t. This updating rule can be rewritten as

$$\boldsymbol{\mu}_{k^*}^{(t+1)} = \boldsymbol{\mu}_{k^*}^{(t)} + \alpha^{(t)}(\mathbf{x}_t - \boldsymbol{\mu}_{k^*}^{(t)}) \tag{2.29}$$

where $\alpha^{(t)} = \frac{(\#\text{class } k^*)^{(t)}}{(\#\text{class } k^*)^{(t)}+1}$ is the learning rate. It can be seen that this learning rule is an example of the winner-take-all (WTA) rule in competitive learning (Kohonen, 2001; Kong and Kosko, 1991), because only the cluster that "wins" the data point can be updated from it.

2.3.2 Model-based clustering

In the previous section we presented the main common partition-based clustering algorithm, that is K-means. In this section, we describe general clustering methods based on finite mixture models. This approach is known as the model-based clustering (Banfield and Raftery, 1993; Celeux and Govaert, 1993; Fraley and Raftery, 1998, 2002; McLachlan and Basford, 1988; Scott and Symons, 1971).

Clustering via finite mixture models

In the finite mixture approach for cluster analysis, the data probability density function is assumed to be a mixture density, each component density being associated with a cluster. The problem of clustering therefore becomes the one of estimating the parameters of the assumed mixture model (e.g, estimating the mean vector and the covariance matrix for each component density in the case of Gaussian mixtures). In this way, two main approaches are possible. The former is refereed to as the mixture approach or the estimation approach and the latter is known as the classification approach.

The mixture approach In the mixture approach, the parameters of the mixture density are estimated in a maximum likelihood estimation framework generally via the EM algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997). After performing the probability density estimation, the posterior probabilities (c.f., equation (2.9) are then used to determine the cluster memberships through the MAP principle.

The classification approach The classification approach consists of optimizing *classification likelihood* criteria (Scott and Symons, 1971, 1981). More particularly, in the maximum likelihood classification approach, the optimized classification likelihood function is the complete-data likelihood (c.f., Equation (2.7)) and the used optimization algorithm in this case is the CEM algorithm (Celeux and Govaert, 1992). According to the classification approach, the cluster memberships and the model parameters are estimated simultaneously as the learning proceeds (c.f., section 2.3.2). For a more detailed account of both the mixture and the classification approach, the reader is refereed to the paper of Celeux and Govaert (1992). In the next section we describe the CEM algorithm.

Classification EM algorithm (CEM)

The EM algorithm computes the maximum likelihood (ML) estimate of a mixture model. The Classification EM (CEM) algorithm (Celeux and Govaert, 1992) estimates both the mixture model parameters and the classes' labels by maximizing the completed-data log-likelihood $\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{X},\mathbf{z}) = \log p(\mathbf{X},\mathbf{z};\boldsymbol{\Psi})$ (Celeux and Govaert, 1992).

After starting with an initial parameter $\boldsymbol{\Psi}^{(0)}$, the CEM algorithm, in its general formulation, iteratively alternates between the two following steps until convergence:

Step 1: Compute the missing data $\mathbf{z}^{(q+1)}$ given the observations and the current estimated model parameters $\boldsymbol{\Psi}^{(q)}$:

$$\mathbf{z}^{(q+1)} = \arg\max_{\mathbf{z}\in\mathcal{Z}^n} \mathcal{L}_c(\boldsymbol{\Psi}^{(q)};\mathbf{X},\mathbf{z}) \tag{2.30}$$

Step 2: Compute the model parameters update $\boldsymbol{\Psi}^{(q+1)}$ by maximizing the complete-data log-likelihood given the current estimation of the missing data $\mathbf{z}^{(q+1)}$:

$$\boldsymbol{\Psi}^{(q+1)} = \arg\max_{\boldsymbol{\Psi}\in\boldsymbol{\Omega}} \mathcal{L}_c(\boldsymbol{\Psi};\mathbf{X},\mathbf{z}^{(q+1)}). \tag{2.31}$$

As shown by Celeux and Govaert (1992), the CEM algorithm, for the case of mixture models, is equivalent to integrating a classification step (C-step) between the E- and the M- steps of the EM algorithm. The C-step assigns the observations to the component densities by using the MAP rule. Thus, an iteration of CEM for the case of of finite mixture model can be summarized as follows:

E-step: Compute the conditional posterior probabilities $\tau_{ik}^{(q)}$ that the observation $\mathbf{x}_i$ arises from the kth component density.

C-step: Assign each observation $\mathbf{x}_i$ to the component maximizing the conditional posterior probability τ_{ik}:

$$z_i^{(q+1)} = \arg\max_{k\in\mathcal{Z}} \tau_{ik}^{(q)} \quad (i = 1,\ldots,n). \tag{2.32}$$

Therefore this step provides a hard partition of the data and then the completed-data log-likelihood can be computed based on the obtained partition.

M-step: Update the mixture model parameters by maximizing the completed-data log-likelihood for the partition provided by the C-step.

The pseudo code 2 summarizes the CEM algorithm for the case of mixture of Gaussians.

Algorithm 2 Pseudo code of the CEM algorithm for Gaussian mixture models.

Inputs: a data set $\mathbf{X}$ and the number of clusters K

```
 1: fix a threshold $\epsilon > 0$
 2: set $q \leftarrow 0$ (iteration)
 3: Initialize: $\mathbf{\Psi}^{(0)} = (\pi_1^{(0)}, \ldots, \pi_K^{(0)}, \mathbf{\Psi}_1^{(0)}, \ldots, \mathbf{\Psi}_K^{(0)})$ with $\mathbf{\Psi}_k^{(0)} = (\boldsymbol{\mu}_K^{(0)}, \mathbf{\Sigma}_K^{(0)})$
 4: while increment in the complete-data log-likelihood $> \epsilon$ do
 5:   E-step:
 6:   for $k = 1, \ldots, K$ do
 7:     Compute $\tau_{ik}^{(q)}$ for $i = 1, \ldots, n$ using Equation (2.19)
 8:   end for
 9:   C-step:
10:   for $k = 1, \ldots, K$ do
11:     Compute $z_i^{(q)} = \arg\max_{k \in \mathcal{Z}} \tau_{ik}^{(q)}$ for $i = 1, \ldots, n$
12:     Set $z_{ik}^{(q)} = 1$ if $z_i^{(q)} = k$ and $z_{ik}^{(q)} = 0$ otherwise, for $i = 1, \ldots, n$
13:   end for
14:   M-step:
15:   for $k = 1, \ldots, K$ do
16:     Set $\tau_{ik}^{(q)} = z_{ik}^{(q)}$ for $i = 1, \ldots, n$
17:     Compute $\pi_k^{(q+1)}$ using Equation (2.14)
18:     Compute $\boldsymbol{\mu}_k^{(q+1)}$ using Equation (2.20)
19:     Compute $\mathbf{\Sigma}_k^{(q+1)}$ using Equation (2.21)
20:   end for
21:   $q \leftarrow q + 1$
22: end while
```

Output: $\hat{\mathbf{\Psi}} = \mathbf{\Psi}^{(q)}$

$\hat{z}_i = z_i^{(q)}$ $(i = 1, \ldots, n)$

The CEM algorithm is easy to implement, typically faster to converge than EM and monotonically improves the complete-data log-likelihood as the learning proceeds. In addition, the CEM converges toward a local maximum of the complete-data log-likelihood (Celeux and Govaert, 1992). However, it can be noted that CEM provides biased estimates of the mixture model parameters. Indeed, CEM updates the model parameters from a truncated sample contrary to EM for which the model parameters are updated from the whole data through the fuzzy posterior probabilities and therefore the parameter estimations provided by EM are more accurate.

It can be shown that CEM which is formulated in a probabilistic framework, generalizes several algorithms based on distance criteria such as the well-known K-means algorithm (Celeux and Govaert, 1992).

The idea of using model-based clustering is advantageous compared to many other clustering methods. Indeed, many clustering methods are heuristic and impose a fixed structure on the clusters. For example using K-means method tends to produce same size spherical clusters, which is not usually the case in practice where one may need some flexibility in specifying the form of the clusters. Another problem is that the number of clusters must be specified such as in K-means. As it will be derived in the next section, the model-based clustering framework allows for specifying the form of the clusters for producing different statistical models for the data. Determining the number of clusters can be accomplished by using some model selection criteria basing on the optimized likelihood functions.

Parsimonious Gaussian mixture models

Parsimonious Gaussian mixture models (Banfield and Raftery, 1993; Celeux and Govaert, 1995) are statistical models that allow for capturing a specific cluster shapes (e.g., clusters having the same shape or different shapes, spherical or elliptical clusters, etc). This is accomplished by introducing particular decompositions of the covariance matrices for the Gaussian mixture model. Formally, the kth covariance matrix of a Gaussian mixture is decomposed using the following eigenvalue decomposition:

$$\mathbf{\Sigma}_k = \lambda_k \mathbf{D}_k \mathbf{A}_k \mathbf{D}_k^T \tag{2.33}$$

where

- λ_k represents the volume of the kth cluster or component density. We note that the size of the cluster is the number of observations belonging to the cluster, while the volume is the amount of space of the cluster.
- $\mathbf{D}_k$ is a matrix with columns corresponding to the eigenvectors of $\mathbf{\Sigma}_k$ that determines the orientation of the cluster.
- $\mathbf{A}_k$ is a diagonal matrix, whose diagonal entries are the normalized eigenvalues of $\mathbf{\Sigma}_k$ arranged in a decreasing order and its determinant is 1. This matrix is associated with the shape of the cluster.

This eigenvalue decomposition provides three main families of models: the spherical family, the diagonal family, and the general family and produces 14 different models, according to the choice of the configuration for the parameters λ_k, $\mathbf{A}_k$, and $\mathbf{D}_k$ (Celeux and Govaert, 1995). Celeux and Govaert (1995) provide covariance matrix update equations based on these models by using the EM algorithm. Some of these models have a closed form, and others must be solved in an iterative manner.

In addition to providing flexible statistical models for the clusters, parsimonious Gaussian mixture can be viewed as techniques for reducing the number of parameters in the model. Indeed, for the general Gaussian mixture model, the EM algorithm may considerably become slow in the case of high dimensional data. Therefore, imposing constraints on the covariance matrices reduces the dimension of the optimization problem. The EM algorithms therefore provide more accurate estimations compared to the full mixture model.

2.3.3 Assessing the number of clusters: model selection

The problem of choosing the number of clusters can be seen as a model selection problem. The model selection task consists of choosing a suitable compromise between flexibility so that a reasonable fit to the available data is obtained, and over-fitting. A common way is to use a criterion (score function) that ensure the compromise. In general, we choose an overall score function that is explicitly composed of two components: a component that measures the goodness of fit of the model to the data, and a penalty component that governs the model complexity. This yields an overall score function of the form

$$\text{score(model)} = \text{error(model)} + \text{penalty(model)}$$

which will be minimized.

As in general the complexity of a model $\mathcal{M}$ is related to the number of its (free) parameters ν, the penalty function then involves the number of model parameters. In this section, we cite the most commonly used criteria for model selection in a probabilistic modeling framework. Let $\mathcal{M}$ denote a model, $\mathcal{L}(\hat{\boldsymbol{\Psi}})$ its log-likelihood and ν the number of its free parameters. Consider that we fitted M different model structures $(\mathcal{M}_1, \ldots, \mathcal{M}_M)$, from which we wish to choose the "best" one (ideally the one providing the best prediction on future data). Assume we have estimated the model parameters $\hat{\boldsymbol{\Psi}}_m$ for each model structure $\mathcal{M}_m$ $(m = 1, \ldots, M)$ from a sample of n observations $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ and now we wish to choose among these fitted models. The widely used information criteria are defined as follows[1]:

- Akaike Information Criterion (AIC) (Akaike, 1974):

$$\text{AIC}(\mathcal{M}_m) = \mathcal{L}(\hat{\boldsymbol{\Psi}}_m) - \nu_m \tag{2.34}$$

- AIC3 (Bozdogan, 1983):

$$\text{AIC3}(\mathcal{M}_m) = \mathcal{L}(\hat{\boldsymbol{\Psi}}_m) - \frac{3\nu_m}{2} \tag{2.35}$$

- Bayesian Information Criterion (BIC) (Schwarz, 1978):

$$\text{BIC}(\mathcal{M}_m) = \mathcal{L}(\hat{\boldsymbol{\Psi}}_m) - \frac{\nu_m \log(n)}{2} \tag{2.36}$$

[1]Note that here these criteria are given in such a way we attempt to maximize them (this is done done by just adding a minus sign to the standard formulation in which the criteria are rather minimized).

- Integrated Classification Likelihood (ICL)(Biernacki et al., 2000):

$$\mathrm{ICL}(\mathcal{M}_m) = \mathcal{L}_c(\hat{\boldsymbol{\Psi}}_m) - \frac{\nu_m \log(n)}{2} \tag{2.37}$$

where $\mathcal{L}_c(\hat{\boldsymbol{\Psi}}_m)$ is the complete-data log-likelihood for the model $\mathcal{M}_m$ for the complete data $(\mathbf{X}, \mathbf{z}) = ((\mathbf{x}_1, z_1), \ldots, (\mathbf{x}_n, z_n))$, and ν_m denotes the number of free model parameters. For example, in the case of a d-dimensional Gaussian mixture model we have:

$$\nu = \underbrace{(K-1)}_{\pi_k\text{'s}} + \underbrace{K \times d}_{\{\boldsymbol{\mu}_k\}} + \underbrace{K \times \frac{d \times (d+1)}{2}}_{\{\boldsymbol{\Sigma}_k\}} = \frac{K \times (d+1) \times (d+2)}{2} - 1.$$

In the case of selecting the number of mixture components for a mixture model, Celeux and Soromenho (Celeux and Soromenho, 1993, 1996) proposed an entropy criterion called NEC (Normalized Entropy Criterion). This criterion is derived from a particular decomposition of the log-likelihood function. Consider the log-likelihood for a mixture model that is given by Equation (2.4), and denote it by $\mathcal{L}(K)$ (to refer to the fact that it is computed for a mixture of K component densities), it can be shown that the following decomposition holds

$$\mathcal{L}(K) = C(K) + E(K)$$

where

$$C(K) = \sum_{k=1}^{K}\sum_{i=1}^{n} \tau_{ik} \log[\pi_k f(\mathbf{x}_i; \boldsymbol{\Psi}_k)]$$

which represents a classification log-likelihood term and

$$E(K) = -\sum_{k=1}^{K}\sum_{i=1}^{n} \tau_{ik} \log \tau_{ik} \geq 0$$

which is an entropy term of the fuzzy classification defined by the posterior probabilities

$$\tau_{ik} = \frac{\pi_k f(\mathbf{x}_i; \boldsymbol{\Psi}_k)}{\sum_{\ell=1}^{K} \pi_\ell f(\mathbf{x}_i; \boldsymbol{\Psi}_\ell)}$$

and which measures the degree of overlap between the clusters. The NEC criterion is given by:

$$\mathrm{NEC} = \frac{E(K)}{\mathcal{L}(1) - \mathcal{L}(K)}. \tag{2.38}$$

However, it can be seen from the previous equation that NEC(1) (for one cluster) is not defined. An extension that deals with this problem was presented by Biernacki et al. (1999).

Figure 2.4 shows an illustration for clustering using the re-scaled two-dimensional data Old Faithful geyser data set[1].

[1]These data represent the waiting times (in minutes) between eruptions of the Old Faithful geyser at Yellowstone (Hand et al., 1994)

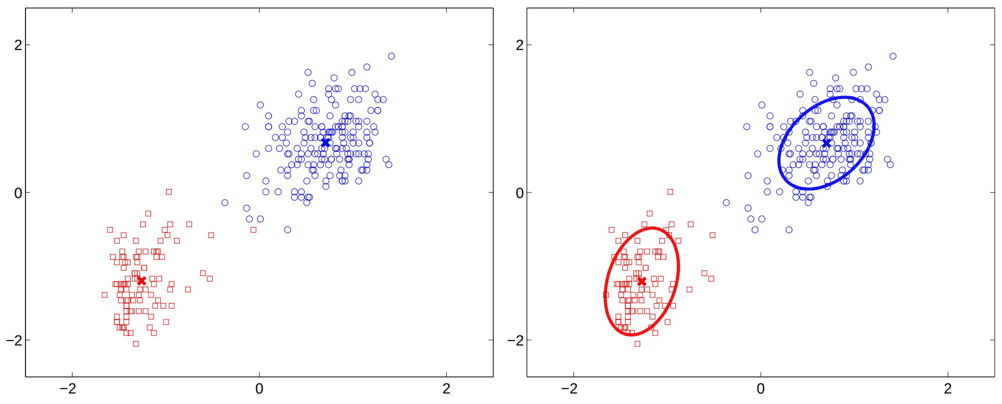


Figure 2.4: Clustering results obtained with K-means algorithm (left) with $K = 2$ and the EM algorithm (right). The cluster centers are shown by the red and blue crosses and the ellipses are the contours of the Gaussian component densities at level 0.4 estimated by EM. The number of clusters for EM have been chosen by BIC for $K = 1, \ldots, 4$.

2.4 Multidimensional data classification

Until now we considered the problem of unsupervised learning where the class labels are unknown (hidden). When the class labels are given, the tasks of class prediction for new observations (inputs) consists of supervised learning for classification. In this section we will describe some well-known and widely used approaches for multidimensional data classification.

Given a training data set comprising n labeled observations $((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n))$ where $\mathbf{x}$ denotes the observation (or the input) which is assumed to be continuous-valued in $\mathcal{X} = \mathbb{R}^d$ and y denotes the target variable (or the output) representing the class label which is a discrete-valued variable in $\mathcal{Y} = \{1, \ldots, G\}$, G being the number of classes. In this context of classification, the aim is to predict the value of the class label y for a new observation $\mathbf{x}$. Two common approaches based on generative classifiers and discriminative classifiers, may be used (Jebara, 2001, 2003). Discriminative classifiers learn a direct map from the inputs $\mathbf{x}$ to the class labels y or directly learn a model of the conditional distribution $p(y|\mathbf{x})$. From this conditional distribution we can make predictions of y, for any new value of $\mathbf{x}$, by using the MAP rule:

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} p(y|\mathbf{x}). \tag{2.39}$$

Generative classifiers learn a model of the joint distribution $p(\mathbf{x}, y)$ which consist in modeling the class conditional densities given by $p(\mathbf{x}|y)$, together with the prior probabilities $p(y)$ for $y = 1, \ldots, G$. The posterior class probabilities are then computed using Bayes' theorem:

$$p(y|\mathbf{x}) = \frac{p(y)p(\mathbf{x}|y)}{\sum_{y'} p(y')p(\mathbf{x}|y')}. \tag{2.40}$$

In the following we give an overview of probabilistic approaches for multidimensional data modeling and classification.

2.4.1 Linear Discriminant Analysis

Consider the problem of classification where the aim is to classify the observed variables into G predefined groups or classes. Typical generative approaches consist of modeling each conditional-class density by a multivariate Gaussian density:

$$p(\mathbf{x}|y=g;\mathbf{\Psi}_g) = \frac{1}{(2\pi)^{\frac{d}{2}}|\mathbf{\Sigma}_g|^{\frac{1}{2}}}\exp\Big(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_g)^T\mathbf{\Sigma}_g^{-1}(\mathbf{x}-\boldsymbol{\mu}_g)\Big) \tag{2.41}$$

where $\boldsymbol{\mu}_g \in \mathbb{R}^d$ is the mean vector, $\mathbf{\Sigma}_g \in \mathbb{R}^{d\times d}$ is the covariance matrix and $\mathbf{\Psi}_g = (\boldsymbol{\mu}_g, \mathbf{\Sigma}_g)$ for $g = 1,\ldots,G$.

Linear Discriminant Analysis (LDA) arises when we assume that all the classes have a common covariance matrix $\mathbf{\Sigma}_g = \mathbf{\Sigma}\ \forall g = 1,\ldots,G$. The term "linear" in LDA is due to the fact that the decision boundaries between each pair of classes g and h are linear. In fact, in this generative approach, each observation $\mathbf{x}$ is assigned to the class $\hat{y}$ by using the Maximum A Posteriori (MAP) rule:

$$\begin{aligned}\hat{y} &= \arg\max_{g\in\mathcal{Y}} p(y=g|\mathbf{x};\mathbf{\Psi}_g)\\ &= \arg\max_{g\in\mathcal{Y}} \frac{w_g\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_g,\mathbf{\Sigma})}{\sum_{h=1}^G w_h\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_h,\mathbf{\Sigma})}\end{aligned} \tag{2.42}$$

where $w_g = p(y=g)$ is the prior probability of the class g. The decision boundary between classes g and h, which is the set of inputs $\mathbf{x}$ verifying $p(y=g|\mathbf{x}) = p(y=h|\mathbf{x})$, or by equivalence:

$$\begin{aligned}\log\frac{p(y=g|\mathbf{x};\mathbf{\Psi}_g)}{p(y=h|\mathbf{x};\mathbf{\Psi}_h)} = 0 &\Leftrightarrow \log\frac{w_g}{w_h} + \log\frac{\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_g,\mathbf{\Sigma})}{\mathcal{N}(\mathbf{x};\boldsymbol{\mu}_h,\mathbf{\Sigma})} = 0\\ &\Leftrightarrow \log\frac{w_g}{w_h} - \frac{1}{2}(\boldsymbol{\mu}_g+\boldsymbol{\mu}_h)^T\mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_g-\boldsymbol{\mu}_h) + \mathbf{x}^T\mathbf{\Sigma}^{-1}(\boldsymbol{\mu}_g-\boldsymbol{\mu}_h) = 0,\end{aligned} \tag{2.43}$$

which is a linear function in $\mathbf{x}$ and therefore the classes will be separated by hyperplanes in the input space.

Each of the class prior probabilities w_g is calculated with the proportion of the class g in the training data set and is given by

$$w_g = \frac{\sum_{i|y_i=g}}{n} = \frac{n_g}{n}. \tag{2.44}$$

where n_g is the cardinal number of class g. The unknown parameters $\mathbf{\Psi}_g$ for each class g are estimated by maximum likelihood given a labeled training set. Given and independent and identically distributed (i.i.d) sample, the log-likelihood of $\mathbf{\Psi}_g$ is given by:

$$\mathcal{L}(\mathbf{\Psi}_g) = \log\prod_{i|y_i=g}\mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_g,\mathbf{\Sigma}) = \sum_{i|y_i=g}\log\mathcal{N}(\mathbf{x}_i;\boldsymbol{\mu}_g,\mathbf{\Sigma}). \tag{2.45}$$

The problem can be solved in a closed form. Calculating the derivative of $\mathcal{L}(\mathbf{\Psi}_g)$ with respect to $\boldsymbol{\mu}_G$ and $\boldsymbol{\Sigma}_g$ and setting it to zero yields:

$$\hat{\boldsymbol{\mu}}_g = \frac{1}{n_g} \sum_{i|y_i=g} \mathbf{x}_i, \tag{2.46}$$

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n-G} \sum_{g=1}^{G} \sum_{i|y_i=g} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_g)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_g)^T, \tag{2.47}$$

2.4.2 Quadratic Discriminant Analysis

Quadratic Discriminant Analysis (QDA) is an extension of LDA that considers a different covariance matrix for each class. As in LDA, the decision functions between two classes g and h are computed from the log-ratio of the posterior probabilities of the class g and the class h:

$$\begin{aligned}\log \frac{p(y=g|\mathbf{x})}{p(y=h|\mathbf{x})} &= \log \frac{w_g}{w_h} - \frac{1}{2} \log \frac{|\boldsymbol{\Sigma}_g|}{|\boldsymbol{\Sigma}_h|} \\ &\quad - \frac{1}{2}\{(\mathbf{x} - \boldsymbol{\mu}_g)^T \boldsymbol{\Sigma}_g^{-1} (\mathbf{x} - \boldsymbol{\mu}_g) - (\mathbf{x} - \boldsymbol{\mu}_h)^T \boldsymbol{\Sigma}_h^{-1} (\mathbf{x} - \boldsymbol{\mu}_h)\} = 0. \end{aligned} \tag{2.48}$$

This function is quadratic in $\mathbf{x}$, we then get quadratic discriminant functions in the input space. The unknown parameters $\boldsymbol{\Psi}_g$ for QDA are estimated similarly as for LDA, except that separate covariance matrices must be estimated for each class. The estimations are given by:

$$\hat{\boldsymbol{\mu}}_g = \frac{1}{n_g} \sum_{i|y_i=g} \mathbf{x}_i \tag{2.49}$$

$$\hat{\boldsymbol{\Sigma}}_g = \frac{1}{n_g} \sum_{i|y_i=g} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_g)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_g)^T. \tag{2.50}$$

2.4.3 Mixture Discriminant Analysis

We have seen for the Gaussian case, in both LDA and QDA, that each class density is modeled by a single Gaussian density. This may be limited for modeling non homogeneous classes where the classes are dispersed. In Mixture Discriminant Analysis (MDA) (Hastie and Tibshirani, 1996), each class density is modeled by a Gaussian mixture density (McLachlan and Peel., 2000; Titterington et al., 1985). The MDA approach can therefore capture many specific properties of real data such as multimodality, unobserved heterogeneity, heteroskedasticity, etc. For MDA, the Gaussian mixture density is defined by:

$$p(\mathbf{x}|y=g;\boldsymbol{\Psi}_g) = \sum_{r=1}^{R_g} \pi_{gr} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{gr}, \boldsymbol{\Sigma}_{gr}) \tag{2.51}$$

where R_g is the number of mixture components for class g,

$$\boldsymbol{\Psi}_g = (\pi_{g1}, \ldots, \pi_{gR_g}, \boldsymbol{\mu}_{g1}, \ldots, \boldsymbol{\mu}_{gR_g}, \ldots, \boldsymbol{\Sigma}_{g1}, \ldots, \boldsymbol{\Sigma}_{gR_g})$$

is the parameter vector of the mixture density of class g and the π_{gr}'s $(r = 1, \ldots, R_g)$ are the non-negative mixing proportions satisfying $\sum_{r=1}^{R_g} \pi_{gr} = 1 \; \forall g$. Note that here we allow a different covariance matrix for each mixture mixture component. However, one can also consider a mixture model where the component densities have a common covariance matrix (Hastie and Tibshirani, 1996).

Figure 2.5 shows a three-class example in $\mathbb{R}^2$.

The classification methods described in the previous section are based on probabilistic generative methods. In the next section we will discuss a widely used probabilistic discriminative method for classification: the logistic regression.

2.4.4 Multi-class Logistic Regression

Logistic regression is a probabilistic discriminative approach widely used in supervised learning (Hastie et al., 2010; Hosmer and Lemeshow, 2000). It is widely used for binary classification, for example in diagnosis problems: patients have disease or not, etc.

The model

The multi-class logistic regression model directly models the classes' posterior probabilities via the following model[1]:

$$p(y = g|\mathbf{x}) = \pi_g(\mathbf{x}; \mathbf{w}) = \frac{\exp(\boldsymbol{w}_g^T \mathbf{x})}{\sum_{h=1}^{G} \exp(\boldsymbol{w}_h^T \mathbf{x})} \tag{2.52}$$

for $g = 1, \ldots, G$, which is a logistic transformation of a linear functions in $\mathbf{x}$ that ensures that the posterior probabilities are constrained to sum to one and remain in $[0, 1]$. The vector $\mathbf{w} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_G)^T$ represents the model parameter vector which is in $\mathbb{R}^{G \times (d+1)}$. Since the logistic probabilities sum to one, $\boldsymbol{w}_G$ is generally set to the null vector to ensure this constraint. The learning procedure of the model is given in the next section.

Parameter estimation

Here we will discuss in detail the multi-class case (the binary case being a particular case of the general multinomial setting). The maximum likelihood is generally used to fit the model (Minka, 2001). Suppose we have a labeled independent sample $((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n))$. The conditional log-likelihood of $\mathbf{w}$ for the given class labels

[1]Notice that in the model (2.52) we add a constant coordinate (1) to the input vector $\mathbf{x}$ so that, hereafter, for logistic regression we have $\mathbf{x} = (x_1, \ldots, x_d, 1)^T$ which is in $\mathbb{R}^{d+1}$. This choice is used to simplify the mathematical expressions, namely for the derivatives of the logistic function.

$\mathbf{y} = (y_1, \ldots, y_n)$ conditionally on the inputs $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ can be written as

$$\begin{aligned}\mathcal{L}(\mathbf{w}) = \mathcal{L}(\mathbf{w}; \mathbf{X}, \mathbf{y}) &= \log \prod_{i=1}^{n} p(y_i|\mathbf{x}_i; \mathbf{w}) \\ &= \log \prod_{i=1}^{n} \prod_{g=1}^{G} p(y_i = g|\mathbf{x}_i; \mathbf{w})^{y_{ig}} \\ &= \sum_{i=1}^{n} \sum_{g=1}^{G} y_{ig} \log \pi_g(\mathbf{x}_i; \mathbf{w}) \end{aligned} \tag{2.53}$$

where y_{ig} is an indicator binary variable such that $y_{ig} = 1$ if and only $y_i = g$ (i.e, $\mathbf{x}_i$ belongs to the class g). This log-likelihood can not be maximized in a closed form. The Newton-Raphson (NR) algorithm is generally used to perform the maximization as well as other gradient-based techniques (see (Minka, 2001)). Here we consider the use of the Newton-Raphson algorithm to maximize $\mathcal{L}(\mathbf{w})$ with respect to $\mathbf{w}$.

The Newton-Raphson algorithm is an iterative numerical optimization algorithm which consists in starting with an initial arbitrary solution $\mathbf{w}^{(0)}$, and updating the estimation of $\mathbf{w}$ until a convergence criterion is reached (typically, in this case, when the relative variation of $\mathcal{L}(\mathbf{w})$ is below a prefixed threshold). A single NR update is given by:

$$\mathbf{w}^{(l+1)} = \mathbf{w}^{(l)} - \left[\frac{\partial^2 \mathcal{L}(\mathbf{w})}{\partial \mathbf{w} \partial \mathbf{w}^T}\right]^{-1} \frac{\partial \mathcal{L}(\mathbf{w})}{\partial \mathbf{w}} \tag{2.54}$$

where the Hessian and the gradient of $\mathcal{L}(\mathbf{w})$ (which are respectively the second and first derivative of $\mathcal{L}(\mathbf{w})$) are evaluated at $\mathbf{w} = \mathbf{w}^{(l)}$.

The gradient component $\frac{\partial \mathcal{L}(\mathbf{w})}{\partial \boldsymbol{w}_h}$ $(h = 1, \ldots, G-1)$ is given by (see Appendix A.3:

$$\frac{\partial \mathcal{L}(\mathbf{w})}{\partial \boldsymbol{w}_h} = \sum_{i=1}^{n} \big(y_{ih} - \pi_h(\mathbf{x}_i; \mathbf{w})\big)\mathbf{x}_i \tag{2.55}$$

which can be formulated in a matrix form as

$$\frac{\partial \mathcal{L}(\mathbf{w})}{\partial \boldsymbol{w}_h} = \mathbf{X}^T(\mathbf{y}_h - \mathbf{p}_h) \tag{2.56}$$

where $\mathbf{X}$ is the $n \times (d+1)$ matrix whose rows are the input vectors $\mathbf{x}_i$, $\mathbf{y}_h$ is the $n \times 1$ column vector whose elements are the indicator variables y_{ih} for the hth logistic component:

$$\mathbf{y}_h = (y_{1h}, \ldots, y_{nh})^T$$

and $\mathbf{p}_h$ is the $n \times 1$ column vector of logistic probabilities corresponding to the ith input

$$\mathbf{p}_h = (\pi_h(\mathbf{x}_1; \mathbf{w}), \ldots, \pi_h(\mathbf{x}_n; \mathbf{w}))^T.$$

Thus, the matrix formulation of the gradient of $\mathcal{L}(\mathbf{w})$ for all the logistic components is

$$\frac{\partial \mathcal{L}(\mathbf{w})}{\partial \mathbf{w}} = \mathbf{X}^{*T}(\mathbf{Y} - \mathbf{P}) \tag{2.57}$$

where $\mathbf{Y} = (\mathbf{y}_1^T, \ldots, \mathbf{y}_{G-1}^T)^T$ and $\mathbf{P} = (\mathbf{p}_1^T, \ldots, \mathbf{p}_{G-1}^T)^T$ are $n \times (G-1)$ column vectors and $\mathbf{X}^*$ is the $(n \times (G-1))$ by $(d+1)$ matrix of $G-1$ copies of $\mathbf{X}$ such that $\mathbf{X}^* = (\mathbf{X}^T, \ldots, \mathbf{X}^T)^T$. It is therefore easy to verify that the gradient vector is of dimension $(G-1) \times (d+1)$.

The Hessian matrix is composed of $(G-1) \times (G-1)$ block matrices where each block matrix is of dimension $(d+1) \times (d+1)$ and is given by:

$$\frac{\partial^2 \mathcal{L}(\mathbf{w})}{\partial \boldsymbol{w}_h \partial \boldsymbol{w}_k^T} = -\sum_{i=1}^{n} \pi_h(\mathbf{x}_i; \mathbf{w}) \left(\delta_{hk} - \pi_k(\mathbf{x}_i; \mathbf{w})\right) \mathbf{x}_i \mathbf{x}_i^T \tag{2.58}$$

(c.f., Appendix A.3, which can be formulated in a matrix form as

$$\frac{\partial^2 \mathcal{L}(\mathbf{w})}{\partial \boldsymbol{w}_h \partial \boldsymbol{w}_k^T} = -\mathbf{X}^T \mathbf{W}_{hk} \mathbf{X} \tag{2.59}$$

where $\mathbf{W}_{hk}$ is the $n \times n$ diagonal matrix whose diagonal elements are $\pi_h(\mathbf{x}_i; \mathbf{w}) \left(\delta_{hk} - \pi_k(\mathbf{x}_i; \mathbf{w})\right)$ for $i = 1, \ldots, n$. For all the logistic components $(h, k = 1, \ldots, G-1)$, the Hessian takes the following form:

$$\frac{\partial^2 \mathcal{L}(\mathbf{w})}{\partial \mathbf{w} \partial \mathbf{w}^T} = -\mathbf{X}^{*T} \mathbf{W} \mathbf{X}^* \tag{2.60}$$

where $\mathbf{W}$ is the $(n \times (G-1))$ by $(n \times (G-1))$ matrix composed of $(G-1)) \times (G-1))$ block matrices, each block is $\mathbf{W}_{hk}$ $(h, k = 1, \ldots, G-1)$. It can be shown that the Hessian matrix for the multi-class logistic regression model is positive semi definite and therefore the optimized log-likelihood is concave (Bishop, 2006). The NR algorithm (2.54) in this case can therefore be reformulated from the Equations (2.57) and (2.60) as

$$\begin{aligned}
\mathbf{w}^{(l+1)} &= \mathbf{w}^{(l)} + (\mathbf{X}^{*T} \mathbf{W}^{(l)} \mathbf{X}^*)^{-1} \mathbf{X}^{*T} (\mathbf{Y} - \mathbf{P}^{(l)}) \\
&= (\mathbf{X}^{*T} \mathbf{W}^{(l)} \mathbf{X}^*)^{-1} \left[\mathbf{X}^{*T} \mathbf{W}^{(l)} \mathbf{X}^* \mathbf{w}^{(l)} + \mathbf{X}^{*T} (\mathbf{Y} - \mathbf{P}^{(l)})\right] \\
&= (\mathbf{X}^{*T} \mathbf{W}^{(l)} \mathbf{X})^{-1} \mathbf{X}^{*T} \left[\mathbf{W}^{(l)} \mathbf{X}^* \mathbf{w}^{(l)} + (\mathbf{Y} - \mathbf{P}^{(l)})\right] \\
&= (\mathbf{X}^{*T} \mathbf{W}^{(l)} \mathbf{X}^*)^{-1} \mathbf{X}^{*T} \mathbf{W}^{(l)} \mathbf{Y}^*
\end{aligned} \tag{2.61}$$

where $\mathbf{Y}^* = \mathbf{X}^* \mathbf{w}^{(l)} + (\mathbf{W}^{(l)})^{-1} (\mathbf{Y} - \mathbf{P}^{(l)})$ which yields in the Iteratively Reweighted Least Squares (IRLS) algorithm.

The main advantage of the Newton-Raphson method is its quadratic convergence (Boyd and Vandenberghe, 2004).

Illustration

Figure 2.5 shows an illustration of the classification approaches using a synthetic data set comprising three classes. The decision boundaries are shown and it can be observed that that MDA can deal with the problem of heterogeneous classes due to the mixture formulation which provides better class separation due to the complex non-linear decision boundaries.

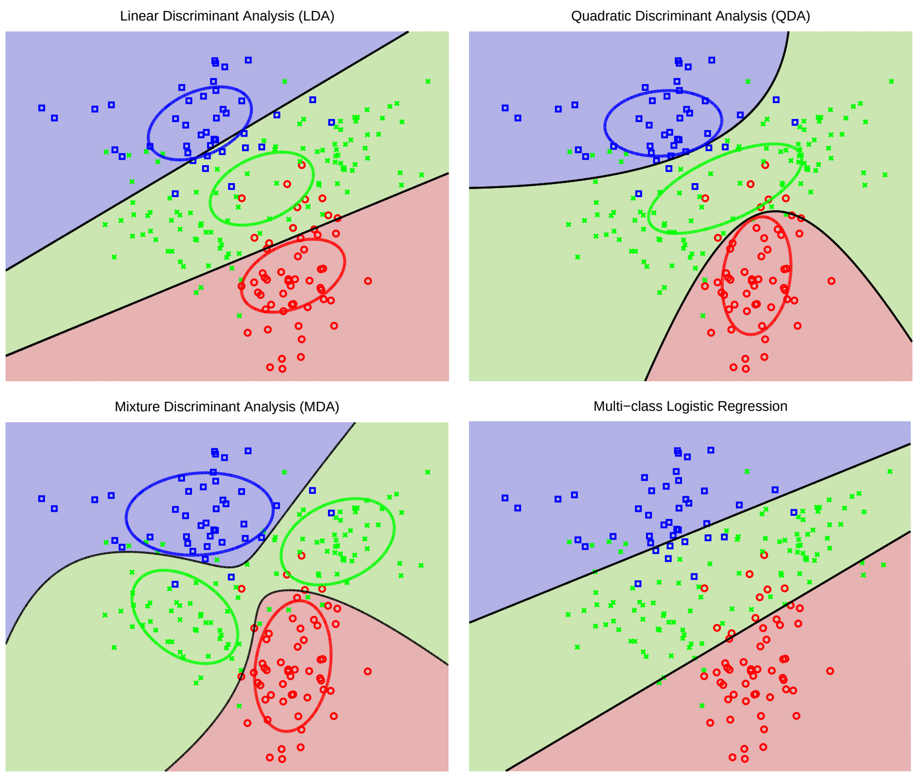


Figure 2.5: A three-class example of a synthetic data set in which one of the classes occurs into two sub-classes, with training data points denoted in blue (□), green (×), and red (○). Ellipses denote the contours of the class probability density functions, lines denote the decision boundaries, and the background colors denote the respective classes of the decision regions. We see that both LDA and Logistic regression provide linear separation, while QDA and MDA provide non linear separation. MDA can further deal the problem of heterogeneous classes.

2.5 General approaches for multidimensional sequential data

Until now we have considered independence assumption for the observations which were assumed to be independent and identically distributed (i.i.d). In this section we will relax this assumption by allowing a dependence between the data which are supposed to be an observation sequence and therefore ordered in the time.

2.5.1 Markov chains

Markov chains are statistical modeling tools used for modeling many phenomena in several application domains. A Markov chain is a sequence of n random variables $(z_1, \ldots, z_n)$, generally referred to as the *states* of the chain, verifying the Markov property that is, the current state given the previous state sequence depends only on the previous state. Formally,

$$p(z_t|z_{t-1}, z_{t-2}, \dots, z_1) = p(z_t|z_{t-1}) \ \forall t > 1. \tag{2.62}$$

The probabilities $p(.|.)$ computed from the distribution p are called the *transition probabilities*. When the transition probabilities do not depend on t, the chain is called a *homogeneous* or *a stationary* Markov chain. The standard Markov chain can be extended by assuming that the current state depends on a history of the state sequence, in this cas one can speak about high order Markov chains (see for example the thesis of Muri (1997)). Formally, a Markov chain of order p, p being a finite integer, can be defined as

$$p(z_t|z_{t-1}, z_{t-2}, \dots, z_1) = p(z_t|z_{t-1}, \dots, z_{t-p}) \ \forall t > p. \tag{2.63}$$

Markov chains are often integrated in a statistical latent data model for sequential data. In this case, the hidden sequence is assumed to be a Markov chain. The resulting model, so-called Hidden Markov Model (HMM) will be introduced in the next section.
.

2.5.2 Hidden Markov Models

Hidden Markov Models (HMMs) are a class of latent data models widely used in many application domains, including speech recognition, image analysis, time series prediction (Derrode and Pieczynski, 2006; Rabiner, 1989), etc. An HMM is a statistical model appropriate for modeling sequential data in which successive samples are no longer assumed to be independent. It can therefore be seen as a generalization of the mixture model by relaxing the independence assumption. Let us denote by $\mathbf{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_n)$ the observation sequence where the multidimensional data example $\mathbf{y}_t$ is observed data at time t, and let us denote by $\mathbf{z} = (z_1, \dots, z_n)$ the hidden state sequence where the discrete random variable z_t which takes its values in the finite set $\mathcal{Z} = \{1, \dots, K\}$ represents the unobserved state associated with $\mathbf{y}_t$. An HMM is fully determined by:

- the initial distribution $\pi = (\pi_1, \dots, \pi_K)$ where $\pi_k = p(z_1 = k); k \in \{1, \dots, K\}$,
- the matrix of transition probabilities $\mathbf{A}$ where $\mathbf{A}_{\ell k} = p(z_t = k|z_{t-1} = \ell)$ for $t = 2, \dots, n$, satisfying $\sum_k \mathbf{A}_{\ell k} = 1$,
- the set of parameters $(\mathbf{\Psi}_1, \dots, \mathbf{\Psi}_K)$ of the parametric conditional probability densities of the observed data $p(\mathbf{y}_t|z_t = k; \mathbf{\Psi}_k)$ for $t = 1, \dots, n$ and $k = 1, \dots, K$. These probabilities are also called the *emission probabilities*.

Figure 2.6 shows a graphical model representation of an HMM with Gaussian emission probabilities.

With this specification of state dependency, the distribution of a particular configuration $\mathbf{z} = (z_1, \dots, z_n)$ of the latent state sequence is written as

$$p(\mathbf{z}; \pi, \mathbf{A}) = p(z_1; \pi) \prod_{t=2}^{n} p(z_t|z_{t-1}; \mathbf{A}), \tag{2.64}$$

and from the conditional independence of the HMM, that is the observation sequence is independent given a particular configuration of the hidden state sequence, the conditional distribution of the observed sequence is therefore written as

$$p(\mathbf{Y}|\mathbf{z};\boldsymbol{\Psi}) = \prod_{t=1}^{n} p(\mathbf{y}_t|z_t;\boldsymbol{\Psi}). \tag{2.65}$$

We then get the following joint distribution (the complete-data likelihood):

$$\begin{aligned} p(\mathbf{Y},\mathbf{z};\boldsymbol{\Psi}) &= p(\mathbf{z};\mathbf{A},\pi)p(\mathbf{Y}|\mathbf{z};\theta) \\ &= p(z_1;\pi)p(\mathbf{y}_1|z_1;\boldsymbol{\Psi})\prod_{t=2}^{n} p(z_t|z_{t-1};\mathbf{A})p(\mathbf{y}_t|z_t;\boldsymbol{\Psi}). \end{aligned} \tag{2.66}$$

2.5.3 Types of Hidden Markov Models

Hidden Markov Models can be classified according to the properties of their hidden Markov chain and also according to the type of the emission state distribution. Homogeneous HMMs are concerned with models for which the hidden Markov chain has a stationary transition matrix. Non-homogeneous HMMs (Diebold et al., 1994; Hughes et al., 1999) arise in the case when a temporal dependency is assumed for the HMM transition probabilities. In this way, Meila and Jordan (1996) also proposed a non-homogeneous HMM as an extension of the mixture of experts (Jacobs et al., 1991). Sometimes, depending on the application, one may aim at modeling a phenomena in which the states proceed from left to right according to the state indexes in a successive manner, for example in speech signals (Rabiner and Juang, 1993). This can be achieved by imposing some restriction for the model through imposing particular constraints on the transition matrix, and yields left-right HMMs (Rabiner and Juang, 1993; Rabiner, 1989) for which the transition matrix would have upper-triangular banded transition matrix. For example, for a 3 state left-right HMM, the transition matrix would have the following form

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & 0 \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{pmatrix}.$$

One can also speak about high order HMMs when the current state depends on a finite history of the HMM states rather than only on the previous one (see for example the thesis of (Muri, 1997)). HMMs have also been extended by integrating additional input variables, in addition of the observations and the hidden sequence, which is particularly adapted for modeling discrete state dynamical systems. This extension is called Input Output HMMs (IOHMMs) and was introduced by Bengio and Frasconi (1995, 1996). Therefore, IOHMMs allow for not only non-stationary transitions between hidden states, but also for modeling output vectors dependent on both hidden states and a set of input variables and can be seen as a probabilistic version for the deterministic finite state machine Bengio and Frasconi (1995). Autoregressive HMM further generalize the standard HMMs by allowing the observations to be Autoregressive Markov chains (Celeux et al., 2004; Frühwirth-Schnatter, 2006; Juang and Rabiner, 1985; Muri, 1997;

Rabiner, 1989). Another HMM extension lies in the Semi-Markov HMM (Murphy, 2002) which is like an HMM except each state can emit a sequence of observations.

Consider the case of HMM with continuous observed variables, the emission probability distribution is this case can be Gaussian and in this case one speaks about Gaussian HMMs. Non-Gaussian HMM have also been introduced. For example one can cite the HMMs with Gaussian mixture emission probabilities (Gauvain and Lee, 1992, 1994; Juang et al., 1985, 1986). Further model extensions can be found in Murphy (2002). In the two next paragraphs we describe two types of HMMs that are HMM with Gaussian emission probabilities and Autoregressive Gaussian HMM.

Gaussian HMM

Consider an HMM with Gaussian emission probabilities. The model can be defined as

$$\mathbf{y}_t = \boldsymbol{\mu}_{z_t} + \boldsymbol{\epsilon}_t \quad ; \quad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{z_t}), \tag{2.67}$$

for $t = 1, \ldots, n$, where the latent sequence $\mathbf{z} = (z_1, \ldots, z_n)$ is drawn from a first-order homogeneous Markov chain and the ϵ_t are independent random variables distributed according to a Gaussian distribution with zero mean and covariance matrix $\boldsymbol{\Sigma}_{z_t}$. According to this model, the state conditional density $p(\mathbf{y}_t|z_t = k; \boldsymbol{\Psi}_k)$ is Gaussian and is given by

$$p(\mathbf{y}_t|z_t = k; \boldsymbol{\Psi}_k) = \mathcal{N}(\mathbf{y}_t; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{2.68}$$

where $\boldsymbol{\Psi}_k = (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$. This Gaussian HMM is illustrated by the graphical structure representation shown in Figure 2.6. The model parameters of an HMM can be learned in

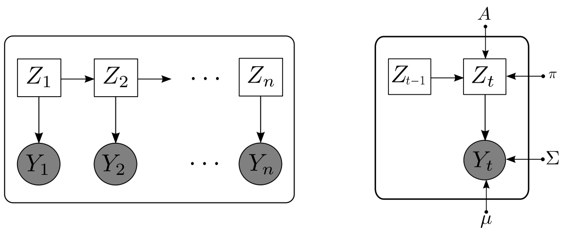


Figure 2.6: Graphical model structure for a Gaussian HMM.

a maximum likelihood framework by the EM algorithm. However, a Bayesian learning can also be adopted (for example see (Gauvain and Lee, 1992, 1994) for the case of learning HMMs with mixture emission probabilities). In the next section we describe the maximum likelihood parameter estimation scheme for the Gaussian HMM.

Parameter estimation

Let $\boldsymbol{\Psi} = (\pi, \mathbf{A}, \boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_K)$ denotes be the model parameter vector to be estimated. The parameter estimation is performed by the maximum likelihood method. The observed-data log-likelihood to be maximized is given by:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Psi};\mathbf{Y}) &= \log p(\mathbf{Y};\boldsymbol{\Psi}) = \log \sum_{\mathbf{z}} p(\mathbf{Y},\mathbf{z};\boldsymbol{\Psi}) \\ &= \log \sum_{z_1,\ldots,z_n} p(z_1;\mathbf{A})p(\mathbf{y}_1|z_1;\boldsymbol{\theta})\prod_{t=2}^{n} p(z_t|z_{t-1};\mathbf{A})p(\mathbf{y}_t|z_t,\boldsymbol{\theta}). \quad (2.69)\end{aligned}$$

Since this log-likelihood is difficult to maximize directly, the EM algorithm (Dempster et al., 1977), known as Baum Welch algorithm (Baum et al., 1970) in the context of HMMs, is generally used to maximize it. With this specification, because of z_t for $t = 1,\ldots,n$ uses a one of K representation, the joint distribution (complete-data likelihood) of a particular configuration $\mathbf{z}$ of the state sequence and the observation sequence $\mathbf{Y}$, (c.f., Equation (2.66)) can be rewritten as

$$\begin{aligned}p(\mathbf{Y},\mathbf{z};\boldsymbol{\Psi}) &= \prod_{k=1}^{K} p(z_1=k;\pi)^{z_{1k}}\prod_{t=2}^{n}\prod_{k=1}^{K}\prod_{\ell=1}^{K} p(z_t=k|z_{t-1}=\ell;\mathbf{A})^{z_{t-1,\ell}z_{tk}}\prod_{t=1}^{n}\prod_{k=1}^{K} p(\mathbf{y}_t|z_t=k;\boldsymbol{\Psi}_k)^{z_{tk}} \\ &= \prod_{k=1}^{K}\pi_k^{z_{1k}}\prod_{t=2}^{n}\prod_{k=1}^{K}\prod_{\ell=1}^{K}\mathbf{A}_{\ell k}^{z_{t-1,\ell}z_{tk}}\prod_{t=1}^{n}\prod_{k=1}^{K} p(\mathbf{y}_t|z_t=k;\boldsymbol{\Psi}_k)^{z_{tk}} \quad (2.70)\end{aligned}$$

where $z_{tk} = 1$ if $z_t = k$ (i.e $\mathbf{y}_t$ originates from the kth state at time t) and $z_{tk} = 0$ otherwise. By taking the logarithm of (2.70) we get the complete-data log-likelihood of $\boldsymbol{\Psi}$:

$$\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{z}) = \sum_{k=1}^{K} z_{1k}\log\pi_k + \sum_{t=2}^{n}\sum_{k=1}^{K}\sum_{\ell=1}^{K} z_{tk}z_{t-1,\ell}\log\mathbf{A}_{\ell k} + \sum_{t=1}^{n}\sum_{k=1}^{K} z_{tk}\log\mathcal{N}(\mathbf{y}_t;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k). \quad (2.71)$$

The EM algorithm starts with an initial parameter $\boldsymbol{\Psi}^{(0)}$ and repeat the following two steps until convergence:

E-step: This step consists of computing the expectation of the complete-data log-likelihood:

$$\begin{aligned}Q(\boldsymbol{\Psi},\boldsymbol{\Psi}^{(q)}) &= \mathbb{E}\Big[\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{z})|\mathbf{Y};\boldsymbol{\Psi}^{(q)}\Big] \\ &= \sum_{k=1}^{K}\mathbb{E}\Big[z_{1k}|\mathbf{Y};\boldsymbol{\Psi}^{(q)}\Big]\log\pi_k + \sum_{t=2}^{n}\sum_{k=1}^{K}\sum_{\ell=1}^{K}\mathbb{E}\Big[z_{tk}z_{t-1,\ell}|\mathbf{Y};\boldsymbol{\Psi}^{(q)}\Big]\log\mathbf{A}_{\ell k} \\ &\quad + \sum_{t=1}^{n}\sum_{k=1}^{K}\mathbb{E}\Big[z_{tk}|\mathbf{Y};\boldsymbol{\Psi}^{(q)}\Big]\log\mathcal{N}(\mathbf{y}_t;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k) \\ &= \sum_{k=1}^{K} p(z_1=k|\mathbf{Y};\boldsymbol{\Psi}^{(q)})\log\pi_k + \sum_{t=2}^{n}\sum_{k=1}^{K}\sum_{\ell=1}^{K} p(z_t=k,z_{t-1}=\ell|\mathbf{Y};\boldsymbol{\Psi}^{(q)})\log\mathbf{A}_{\ell k} \\ &\quad + \sum_{t=1}^{n}\sum_{k=1}^{K} p(z_t=k|\mathbf{Y};\boldsymbol{\Psi}^{(q)})\log\mathcal{N}(\mathbf{y}_t;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k) \\ &= \sum_{k=1}^{K}\tau_{1k}^{(q)}\log\pi_k + \sum_{t=2}^{n}\sum_{k=1}^{K}\sum_{\ell=1}^{K}\xi_{t\ell k}^{(q)}\log\mathbf{A}_{\ell k} + \sum_{t=1}^{n}\sum_{k=1}^{K}\tau_{tk}^{(q)}\log\mathcal{N}(\mathbf{y}_t;\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k) \quad (2.72)\end{aligned}$$

where

- $\tau_{tk}^{(q)} = p(z_t = k|\mathbf{Y};\mathbf{\Psi}^{(q)})\ \forall t = 1,\ldots,n$ and $k = 1,\ldots,K$ is the posterior probability of the state k at time t given the whole observation sequence and the current parameter estimation $\mathbf{\Psi}^{(q)}$. The τ_{tk}'s are also referred to as the *smoothing probabilities*,
- $\xi_{t\ell k}^{(q)} = p(z_t = k, z_{t-1} = \ell|\mathbf{Y};\mathbf{\Psi}^{(q)})\ \forall t = 2,\ldots,n$ and $k,\ell = 1,\ldots,K$ is the joint posterior probability of the state k at time t and the state ℓ at time $t-1$ given the whole observation sequence and the current parameter estimation $\mathbf{\Psi}^{(q)}$.

As shown in the expression of the Q-function, this step requires the computation of the posterior probabilities $\tau_{tk}^{(q)}$ and $\xi_{t\ell k}^{(q)}$. These posterior probabilities are computed by the forward-backward procedures. The forward procedure computes recursively the probabilities

$$\alpha_{tk} = p(\mathbf{y}_1,\ldots,\mathbf{y}_t, z_t = k;\mathbf{\Psi}), \tag{2.73}$$

where α_{tk} is the probability of observing the partial sequence $(\mathbf{y}_1,\ldots,\mathbf{y}_t)$ and ending with the state k at time t. It can be seen that log-likelihood (2.69) can be computed after the forward pass as: $\log p(\mathbf{Y};\mathbf{\Psi}) = \log\sum_{k=1}^K \alpha_{nk}$. The backward procedure computes the probabilities

$$\beta_{tk} = p(\mathbf{y}_{t+1},\ldots,\mathbf{y}_n|z_t = k;\mathbf{\Psi}) \tag{2.74}$$

β_{tk} being the probability of observing the rest of the sequence $(\mathbf{y}_{t+1},\ldots,\mathbf{y}_1)$ knowing that we start with the k at time t. The forward and backward probabilities are computed recursively by the so-called Forward-Backward algorithm (see Appendix A.4 for the details). Notice that in practice, since the recursive computation of the α's and the β's involve repeated multiplications of small numbers which causes underflow problems, their computation is performed using a scaling technique in order to avoid underflow problems. See (Rabiner and Juang, 1993; Rabiner, 1989) for further discussion. The posterior probabilities are then expressed in function of the forward backward probabilities as follows (c.f., Appendix A.5):

$$\tau_{tk}^{(q)} = \frac{\alpha_{tk}^{(q)}\beta_{tk}^{(q)}}{\sum_{k=1}^K \alpha_{tk}^{(q)}\beta_{tk}^{(q)}} \tag{2.75}$$

and

$$\xi_{t\ell k}^{(q)} = \frac{\alpha_{t-1,\ell}^{(q)} p(\mathbf{y}_t|z_t = k;\boldsymbol{\theta}^{(q)})\beta_{tk}^{(q)}\mathbf{A}_{\ell k}^{(q)}}{\sum_{\ell=1}^K\sum_{k=1}^K \alpha_{t-1,\ell}^{(q)} p(\mathbf{y}_t^{(q)}|z_t = k;\mathbf{\Psi})\beta_{tk}^{(q)}\mathbf{A}_{\ell k}^{(q)}}. \tag{2.76}$$

M-step: In this step, the value of the parameter $\mathbf{\Psi}$ is updated by computing the parameter $\mathbf{\Psi}^{(q+1)}$ maximizing the expectation Q with respect to $\mathbf{\Psi}$. The Q-function (2.72) is decomposed as

$$Q(\mathbf{\Psi},\mathbf{\Psi}^{(q)}) = Q_\pi(\pi,\mathbf{\Psi}^{(q)}) + Q_{\mathbf{A}}(\mathbf{A},\mathbf{\Psi}^{(q)}) + \sum_{k=1}^K Q(\mathbf{\Psi}_k,\mathbf{\Psi}^{(q)}) \tag{2.77}$$

with

$$Q_\pi(\pi, \mathbf{\Psi}^{(q)}) = \sum_{k=1}^{K} \tau_{1k}^{(q)} \log \pi_k, \tag{2.78}$$

$$Q_{\mathbf{A}}(\mathbf{A}, \mathbf{\Psi}^{(q)}) = \sum_{t=2}^{n}\sum_{k=1}^{K}\sum_{\ell=1}^{K} \xi_{t\ell k}^{(q)} \log \mathbf{A}_{\ell k}, \tag{2.79}$$

$$Q_{\mathbf{\Psi}_k}(\mathbf{\Psi}, \mathbf{\Psi}^{(q)}) = \sum_{t=1}^{n} \tau_{tk}^{(q)} \log \mathcal{N}\big(\mathbf{y}_t; \boldsymbol{\mu}_k, \mathbf{\Sigma}_k\big). \tag{2.80}$$

The maximization of $Q(\mathbf{\Psi}, \mathbf{\Psi}^{(q)})$ with respect to $\mathbf{\Psi}$ is then performed by separately maximizing $Q_\pi(\pi, \mathbf{\Psi}^{(q)})$, $Q_{\mathbf{A}}(\mathbf{A}, \mathbf{\Psi}^{(q)})$ and $Q_{\mathbf{\Psi}_k}(\mathbf{\Psi}, \mathbf{\Psi}^{(q)})$ $(k = 1, \ldots, K)$. Maximizing Q_π with respect to π subject to $\sum_k \pi_k = 1$ consist of constrained optimized problem which is solved using Lagrange multipliers. The values maximizing $Q_{\mathbf{A}}$ corresponds to the expected number of transitions from state ℓ to state k relative to the expected total number of transitions away from state ℓ (see (Rabiner and Juang, 1993)). Finally, the maximization of $Q_{\mathbf{\Psi}_k}$ with respect to $\boldsymbol{\beta}_k$ for $k = 1, \ldots, K$ consists of a weighted variant of the problem of estimating the parameters of a Gaussian density (as in Equations (2.20) and (2.21)). The updating formulas are given by:

$$\pi_k^{(q+1)} = \tau_{1k}^{(q)} \tag{2.81}$$

$$\mathbf{A}_{\ell k}^{(q+1)} = \frac{\sum_{t=2}^{n} \xi_{tk\ell}^{(q)}}{\sum_{t=2}^{n} \tau_{t\ell}^{(q)}} \tag{2.82}$$

$$\boldsymbol{\mu}_k^{(q+1)} = \frac{1}{\sum_{t=1}^{n} \tau_{tk}^{(q)}} \sum_{t=1}^{n} \tau_{tk}^{(q)} \mathbf{y}_t \tag{2.83}$$

$$\mathbf{\Sigma}_k^{(q+1)} = \frac{1}{\sum_{t=1}^{n} \tau_{tk}^{(q)}} \sum_{t=1}^{n} \tau_{tk}^{(q)} (\mathbf{y}_t - \boldsymbol{\mu}_k^{(q+1)})(\mathbf{y}_t - \boldsymbol{\mu}_k^{(q+1)})^T. \tag{2.84}$$

Autoregressive Hidden Markov Models

Autoregressive Hidden Markov Models (ARHMMs) (Celeux et al., 2004; Frühwirth-Schnatter, 2006; Rabiner, 1989), particularly beneficial for speech processing and recognition (Kenny et al., 1990; Rabiner, 1989), generalize the standard HMM by relaxing the traditional HMM conditional independence assumption. Indeed, ARHMMs assume temporal dependence in the data not only via the transition probability, but also by allowing a dependence of the state output $\mathbf{y}_t$ on past outputs $\mathbf{r}_t = (\mathbf{y}_{t-1}, \ldots, \mathbf{y}_{t-p})$, $p \geq 1$. An autoregressive HMM of order p is illustrated by the graphical model representation shown in Figure 5.3. The model parameter estimation is still performed by maximum likelihood via EM (see for example the paper of Celeux et al. (2004) and the book of Frühwirth-Schnatter (2006)).

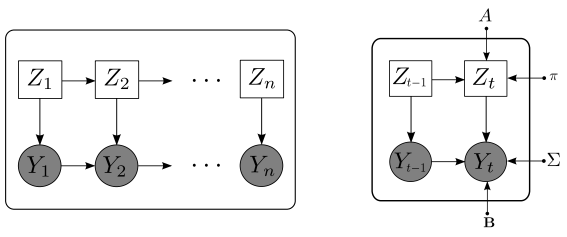


Figure 2.7: Graphical model structure for a Gaussian ARHMM of order 1.

2.6 Curve modeling

Until this stage we have been interested in modeling and learning approaches from multidimensional data. In what follows we will focus on probabilistic modeling and learning from curves (or functional data). We will therefore give an overview of probabilistic approaches for regression, classification and clustering dedicated to curves. These approaches can be associated with the general framework of Functional data Analysis.

2.6.1 Functional data analysis context

Most statistical analyses involve one or more observations taken on each of a number of individuals in a sample, with the aim of making inferences about the general population from which the sample is drawn. In many application domains, these observations are functions (e.g., curves, images). For this reason, statistical methods for analyzing such data are described by the term 'functional data analysis' (FDA) (Ramsay and Dalzell, 1991; Ramsay and Silverman, 2002, 2005). Functional Data Analysis is therefore a general paradigm of data analysis, where the basic unit of information is the entire observed function rather than a finite dimensional vector. The goals of FDA are essentially the same as for other branches of statistics, and include the following: (i) data representation and transformation for further analysis, (ii) data visualization to highlight various characteristics, (iii) explaining variation in an outcome or dependent variable by using input or independent variable, etc. Additional background on FDA, examples and analysis techniques can be found in Ramsay and Silverman (2002) and Ramsay and Silverman (2005). In this thesis, the considered curves are values of functions, available at discretized input time points. Therefore several statistical methods for FDA methods can be used. Particularly, one can cite for example the semi-parametric approach based on Multilayer Perceptrons (MLP) with functional inputs for functional regression and classification introduced by Rossi and Conan-Guez (2005b). The functional MLP have been shown to be universal nonlinear function approximators (Rossi and Conan-Guez, 2006). In this thesis, we will particularly focus on generative regression methods for FDA in the context of curve modeling, classification and clustering.

2.6.2 Polynomial regression and spline regression models

This section is dedicated to regression methods for curve modeling. In what follows we will denote by y the dependent real variable and by t the independent input variable, which represents the time. From a probabilistic perspective, in regression we aim to model the conditional distribution of the output y conditionally on the input t, that is $p(y|t)$:

$$y = f(t) + \epsilon, \tag{2.85}$$

where the function $f(.)$, which is the conditional expectation $\mathbb{E}[y|t]$, can be parametric or non parametric, linear or non linear and ϵ_i is an additive noise, typically a standard Gaussian noise. From the conditional distribution $p(y|t)$ we can make predictions of y, for any new value of t.

Polynomial regression

Polynomial regression models arises when the regression function $f(.)$ is a polynomial function of t. As is generally the case in regression, we assume independent standard Gaussian noise variables ϵ_i. A polynomial regression model can be defined as

$$\begin{aligned} y_j &= f(t_j;\boldsymbol{\beta}) + \sigma\epsilon_j = \sum_{l=0}^{p} \beta_l t_j^l + \sigma\epsilon_j \\ &= \boldsymbol{\beta}^T \mathbf{t}_j + \sigma\epsilon_j; \quad \epsilon_j \sim \mathcal{N}(0,1) \qquad (j = 1,\ldots,m) \end{aligned} \tag{2.86}$$

where the finite integer p represents the order of the polynomial, $\boldsymbol{\beta} = (\beta_0,\ldots,\beta_p)^T$ is the vector of regression coefficients and $\mathbf{t}_j = (1, t_j, t_j^2 \ldots, t_j^p)^T$ is the $p+1$ dimensional covariate vector at time t_j. This model can be reformulated in a matrix form as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{2.87}$$

where $\mathbf{y} = (y_1,\ldots,y_m)^T$, $\boldsymbol{\epsilon} = (\epsilon_1,\ldots,\epsilon_m)^T$ and $\mathbf{X} = (\mathbf{t}_1,\ldots,\mathbf{t}_m)^T$ is the $n \times (p+1)$ regression matrix.

Spline regression

Splines (Deboor, 1978) are widely used for function approximation based on constrained piecewise polynomials. An order-M spline with ordered knots $\zeta_0 < \zeta_1, \ldots, < \zeta_K < \zeta_{K+1}$, including K internal knots, ζ_0 and ζ_{K+1} being the two boundary knots, is a piecewise-polynomial of degree $p = M-1$ with continuous derivatives at the interior knots up to order $M-2$. For example an order-2 spline is a continuous piecewise linear function. The spline regression function can be written as

$$f(t_j) = \sum_{l=0}^{p} \beta_l t_j^l + \sum_{k=1}^{K} \beta_{k+p}(t_j - \zeta_k)_+^p = \boldsymbol{\beta}^T \mathbf{t}_j \tag{2.88}$$

where $(t_j - \zeta_k)_+ = t_j - \zeta_k$ if $t_j \geq \zeta_k$ and $(t_j - \zeta_k)_+ = 0$ otherwise, $\boldsymbol{\beta} = (\beta_0, \ldots, \beta_{K+p})^T$ is the vector of spline coefficients in $\mathbb{R}^{M+K}$ and the corresponding independent covariate vector in this case is given by

$$\mathbf{t}_j = (1, t_j, t_j^2 \ldots, t_j^p, (t_j - \zeta_1)_+^p, \ldots, (t_j - \zeta_K)_+^p).$$

The matrix form of the spline regression model (2.88) can be written in the same manner as for the polynomial regression model (2.86) except the fact that, for the spline regression model, each ith row of the $m \times (M + K)$ regression matrix $\mathbf{X}$ is given by the new covariate vector $\mathbf{t}_j$.

Cubic and natural cubic Splines

The most widely used splines are the cubic splines. A cubic spline is an order-4 spline which corresponds to polynomials of degree $p = 3$. Cubic splines are adapted to higher degrees of smoothness thanks to the continuity of the first and the second derivatives of the piecewise polynomial function at the interior knots.

The piecewise polynomial fitting in the spline regression model can present a bad behavior beyond the boundaries of the region of the inputs t since the constraints of continuity of the piecewise polynomial functions do not concern the regions before the first knot (in $[\zeta_0, \zeta_1]$) and after the last internal knot (in $[\zeta_K, \zeta_{K+1}]$). The natural cubic splines impose additional constraints to deal with this problem by considering that the spline function is linear in these regions. These constraints, that imply second and third derivatives of the spline function in the boundary regions, free up some regression parameters and lead to a particular form of the spline basis function and therefore the spline regression matrix (Hastie et al., 2010).

Smoothing Splines

Smoothing splines (Wahba, 1990) deals with the over-fitting problem which one may have with the spline regression model. Indeed, smoothing splines optimize a penalized function to control the complexity of the spline fit by adding to the optimized criterion a roughness penalty

$$\lambda \int_u (f''(u))^2 du$$

in which the integrated squared second derivative of $f(t)$ measures the curvature of the fitted function. When $\int_u (f''(u))^2 du$ is large, the fitted function is rougher, and when it is small, the fitted function is smoother. The nonnegative smoothing parameter λ is for establishing a trade-off between closeness of fit to the data and a smooth fit. As λ decreases, the fitted function tends to interpolate the data, and we get a rough fit. When λ increases, the result is a smooth fit.

B-splines

For splines, the columns of the regression matrix $\mathbf{X}$ tend to be highly correlated since each column is a transformed version of t. This collinearity may result in a nearly singular matrix and imprecision in the spline fit (Ruppert and Carroll, 2003). B-splines allows for efficient computations thanks to the block matrix form of the regression matrix. An order-M B-spline function is defined as a sum of linear combination of basis functions $B_{l,M}$ as

$$f(t_j) = \sum_{l=1}^{K+M} \beta_l B_{l,M}(t_j),\ t_j \in [\tau_l, \tau_{l+M}] = \boldsymbol{\beta}^T \mathbf{t}_j \tag{2.89}$$

where each Mth order B-spline $B_{l,M}$ is a piecewise polynomial of degree $p = M-1$ that has finite support over $[\tau_l, \tau_{j+M}]$ and is zero everywhere else. Each of the basis functions $B_{l,M}(t_j)$ can be computed as in Appendix B.5 (Hastie et al., 2010) and for the B-spline regression model, each row of the $n \times (M+K)$ B-spline regression matrix $\mathbf{X}$ is given by:

$$\mathbf{t}_j = (B_{1,M}(t_j),\ B_{2,M}(t_j), \ldots,\ B_{M+K,M}(t_j)).$$

Parameter estimation

The parameters $\boldsymbol{\Psi} = (\boldsymbol{\beta}, \sigma^2)$ of the regression models, including polynomial, spline and B-spline regression models, are estimated by maximum likelihood. Given a set of m pairs $(\mathbf{y}, \mathbf{t}) = ((t_1, \mathbf{y}_1), \ldots, (t_m, \mathbf{y}_m))$, the likelihood function of $\boldsymbol{\Psi}$ is given by:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Psi}) &= \log \prod_{j=1}^{m} \mathcal{N}(y_j; \boldsymbol{\beta}^T \mathbf{t}_j, \sigma^2) \\ &= -\frac{1}{2}\Big[\sum_{j=1}^{m} (\frac{y_j - \boldsymbol{\beta}^T \mathbf{t}_j}{\sigma})^2 + m \log \sigma^2\Big] + cst \end{aligned} \tag{2.90}$$

where *cst* is a constant. The solution for $\boldsymbol{\beta}$ is therefore computed by minimizing the corresponding sum of squared error and the estimated coefficient vector $\boldsymbol{\beta}$ is given by:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} \tag{2.91}$$

where the matrix $\mathbf{X}$ differs according to the model. For the smoothing splines, the optimized function corresponds to a penalized sum of least squares which has the same form as the generalized ridge regression (Hastie et al., 2010) and the estimated regression coefficients in this case are given by:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{X} + \lambda \boldsymbol{\Gamma})^{-1} \mathbf{X}^T \mathbf{y}$$

where $\boldsymbol{\Gamma}$ is a square matrix of the same dimension as the vector $\boldsymbol{\beta}$, its elements are the integrated squared second derivatives of $f(.)$.

Finally, the estimation of the variance σ^2 is given by minimizing

$$\frac{1}{\sigma^2} \sum_{j=1}^{m} (y_j - \boldsymbol{\beta}^T \mathbf{t}_j)^2 + m \log \sigma^2$$

w.r.t σ^2 and is given by:

$$\begin{aligned}\hat{\sigma}^2 &= \frac{1}{m}\sum_{j=1}^{m}(y_j - \hat{\boldsymbol{\beta}}^T\mathbf{t}_j)^2 \\ &= \frac{1}{m}(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}})^T(\mathbf{y}-\mathbf{X}\hat{\boldsymbol{\beta}}). \qquad (2.92)\end{aligned}$$

2.6.3 Polynomial regression and spline regression for a set of curves

In this section we aim at estimating a single "mean curve" for a training set of curves rather than a single curve. Let $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ be a training data set of n i.i.d curves where each curve $\mathbf{y}_i$ consists of m measurements (observations) $(y_{i1}, \ldots, y_{im})$ regularly observed at the time points $(t_1, \ldots, t_m)$ for all $i = 1, \ldots, n$. In the following, the term "curves size" will be used to define m.

The general model for a set of curves

The regression models, including polynomial, spline and B-spline regression models seen in section 2.6.2 can be extended to model a set of curves. In this case, the model can be written as

$$y_{ij} = f(t_j;\boldsymbol{\beta}) + \sigma\epsilon_{ij}, \quad \epsilon_{ij} \sim \mathcal{N}(0,1) \qquad (2.93)$$

$\forall i = 1\ldots, n$ and $\forall j = 1, \ldots, m$ where f is the polynomial (respectively spline, B-spline) regression function and $\boldsymbol{\beta}$ is the vector of coefficients and σ is the standard deviation of an independent zero mean Gaussian noise.

Parameter estimation

The parameters $\boldsymbol{\Psi} = (\boldsymbol{\beta}, \sigma^2)$ are estimated by maximum likelihood. The likelihood function of $\boldsymbol{\Psi}$ for a single curve, given the time instants $\mathbf{t} = (t_1, \ldots, t_m)$, is given by:

$$p(\mathbf{y}_i|\mathbf{t};\boldsymbol{\Psi}) = \prod_{j=1}^{m}\mathcal{N}(y_{ij}; f(t_j;\boldsymbol{\beta}), \sigma^2) \qquad (2.94)$$

and the overall log-likelihood given an independent training set $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ of n curves is therefore given by:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Psi}) &= \sum_{i=1}^{n}\sum_{j=1}^{m}\log\mathcal{N}(y_{ij}; f(t_j;\boldsymbol{\beta}), \sigma^2) \\ &= -\frac{1}{2}\Big[\sum_{i=1}^{n}\sum_{j=1}^{m}\Big(\frac{y_{ij}-f(t_j;\boldsymbol{\beta})}{\sigma}\Big)^2 + nm\log\sigma^2\Big] + \text{cst}. \qquad (2.95)\end{aligned}$$

The coefficients vector $\boldsymbol{\beta}$ is then estimated by minimized the corresponding sum of squared errors $\sum_{i=1}^{n}\sum_{j=1}^{m}(y_{ij} - f(t_j;\boldsymbol{\beta}))^2$ and the solution of this least squares problem is given by:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{*T}\mathbf{X}^{*})^{-1}\mathbf{X}^{*T}\mathbf{Y}^{*} \qquad (2.96)$$

where $\mathbf{X}^* = (\mathbf{X}^T, \ldots, \mathbf{X}^T)^T$ is the $nm \times (p+K+1)$ matrix consisting of n copies of the regression matrix $\mathbf{X}$ and $\mathbf{y}^* = (\mathbf{y}_1^T, \ldots, \mathbf{y}_n^T)^T$ is an $nm \times 1$ column vector consisting of all the curves constructed by stacking the column vectors of $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ one after another. For the case of smoothing splines, the solution of the minimized penalized sum of squared errors over all the curves is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{*T}\mathbf{X}^* + \lambda\boldsymbol{\Gamma})^{-1}\mathbf{X}^{*T}\mathbf{y}^* \tag{2.97}$$

where $\boldsymbol{\Gamma}$ is the penalty matrix. Once the coefficients $\boldsymbol{\beta}$ are estimated, one then obtains the fitted function $\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}}$.

Finally, the estimation of the variance σ^2 is obtained by maximizing (2.95) w.r.t σ^2 which is straightforward and is given by:

$$\hat{\sigma}^2 = \frac{1}{nm}(\mathbf{y}^* - \mathbf{X}^{*T}\hat{\boldsymbol{\beta}})^T(\mathbf{y}^* - \mathbf{X}^{*T}\hat{\boldsymbol{\beta}}) \tag{2.98}$$

In summary, splines are based on constrained piecewise polynomial fitting with predefined piecewise locations. Therefore, it should be noticed that in spline regression models (including B-splines), the placement of the knots are generally either fixed by the user or placed uniformly over the range of t_j $(j = 1, \ldots, m)$ and selecting the placement of knots can be a combinatorially complex task. Thus, the spline regression models can not be seen as approaches for automatic detection of regime changes.

Relaxing the regularity constraints for the spline model leads to the standard piecewise polynomial regression model in which the placement of the knots (or the transition points) can be optimized using dynamic programming. However, the resulting approximation is not regular. The next section describes the piecewise polynomial regression model.

2.6.4 Piecewise polynomial regression

Piecewise polynomial regression (Brailovsky and Kempner, 1992; Ferrari-Trecate and Muselli, 2002; Hébrail et al., 2010; McGee and Carleton, 1970; Picard et al., 2007) is a modeling and segmentation method that partitions the curve into K segments, each segment being characterized by its mean polynomial curve and its variance. For this type of modeling, the optimal model parameters are estimated by using a dynamic programming procedure (Bellman, 1961; Stone, 1961) such as Fisher's algorithm (Fisher, 1958). This algorithm optimizes an additive cost function over all the segments of the curve (Brailovsky and Kempner, 1992; Lechevalier, 1990).

The piecewise polynomial regression model

The piecewise polynomial regression model assumes that the curve $\mathbf{y} = (y_1, \ldots, y_m)$ incorporates K polynomial regimes on K intervals whose bounds indexes can be denoted by $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_{K+1})$ with $\zeta_1 = 0$ and $\zeta_{K+1} = m$. This defines a partition of the curve into K polynomial segments of lengths $m_1, \ldots, m_K$: $\{y_j|j \in I_1\}, \ldots, \{y_j|j \in I_K\}$ where $I_k = (\zeta_k, \zeta_{k+1}]$ for $k = 1, \ldots, K$.

Standard polynomial regression models are homoskedastic models as they assume that the different polynomial regression models have the same noise variance (Brailovsky and Kempner, 1992; Ferrari-Trecate and Muselli, 2002; Ferrari-Trecate et al., 2002). However, in our case we shall consider the more general framework of a heteroskedastic model which allows the noise level to vary between the different polynomial regression models. It can be defined as follows:

$$\forall j = 1, \ldots, m, \quad y_j = \begin{cases} \boldsymbol{\beta}_1^T \mathbf{t}_j + \sigma_1 \epsilon_j & \text{if } j \in I_1 \\ \boldsymbol{\beta}_2^T \mathbf{t}_j + \sigma_2 \epsilon_j & \text{if } j \in I_2 \\ \vdots & \\ \boldsymbol{\beta}_K^T \mathbf{t}_j + \sigma_K \epsilon_j & \text{if } j \in I_K \end{cases}, \tag{2.99}$$

where $\boldsymbol{\beta}_k$ is the $(p+1)$-dimensional coefficients vector of a p degree polynomial associated with the kth segment (regime), $\mathbf{t}_j = (1, t_j, t_j^2 \ldots, t_j^p)^T$ is the time dependent $(p+1)$-dimensional covariate vector associated with $\boldsymbol{\beta}_k$ and the $\sigma\epsilon_j$ are independent Gaussian random variables with zero mean and a standard deviation σ, representing the additive noise for each segment k. The matrix formulation of (2.99) can be written as

$$\mathbf{y} = \sum_{k=1}^{K} \mathbf{Z}_k (\mathbf{X}\boldsymbol{\beta}_k + \sigma_k \mathbf{e}), \quad \mathbf{e} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_m), \tag{2.100}$$

where $\mathbf{Z}_k$ is a diagonal matrix whose diagonal elements are the labels of the polynomials regimes $(z_{1k}, \ldots, z_{mk})$, and

$$\mathbf{X} = \begin{bmatrix} 1 & t_1 & t_1^2 & \ldots & t_1^p \\ 1 & t_2 & t_2^2 & \ldots & t_2^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & t_m & t_m^2 & \ldots & t_m^p \end{bmatrix}$$

is the $m \times (p+1)$ regression matrix.

Maximum likelihood estimation for the piecewise polynomial regression model

Let us denote by $(\boldsymbol{\psi}, \boldsymbol{\zeta})$ the piecewise regression parameters where $\boldsymbol{\psi} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \sigma_1^2, \ldots, \sigma_K^2)$ is the set of polynomial coefficients and noise variances, and $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_{K+1})$ the set of indexes of transition time points. Parameter estimation is performed by maximum likelihood. The independence assumption of the noise variables ϵ_{ij} $(j \in I_k)$ involves the independence of y_{ij} $(j \in I_k)$ given the time points t_j $(j \in I_k)$. Thus, according to the model (2.99), it can be easily proved that, within the segment k, the observation y_{ij} given t_j, has a Gaussian distribution with mean $\boldsymbol{\beta}_k^T \mathbf{t}_j$ and variance σ_k^2. The log-likelihood of the parameter vector $(\boldsymbol{\psi}, \boldsymbol{\zeta})$ characterizing the piecewise regression model is the sum of the local log-likelihoods over the K segments that can be written as

$$\begin{aligned}\mathcal{L}(\boldsymbol{\psi},\boldsymbol{\zeta}) &= \log p(\mathbf{y}|\mathbf{t};\boldsymbol{\psi},\boldsymbol{\zeta})\\ &= \sum_{k=1}^{K}\sum_{j\in I_k}\log\mathcal{N}\left(y_j;\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\right).\end{aligned} \tag{2.101}$$

Maximizing this log-likelihood is equivalent to minimizing the criterion

$$J(\boldsymbol{\psi},\boldsymbol{\zeta}) = \sum_{k=1}^{K}\Big[\frac{1}{\sigma_k^2}\sum_{j\in I_k}(y_j-\boldsymbol{\beta}_k^T\mathbf{t}_j)^2 + n_k\log\sigma_k^2\Big] \tag{2.102}$$

with respect to $\boldsymbol{\psi}$ and $\boldsymbol{\zeta}$, where n_k is the number of elements in segment k. Since the criterion J is additive over the K segments, a dynamic programming procedure (Bellman, 1961; Stone, 1961), can be used to perform the global minimization. The dynamic programming will be given in section 2.6.5 which is concerned with the case of a set of n curves. This dynamic procedure has a time complexity of $\mathcal{O}(Kp^2m^2)$ which can be computationally expensive for large sample sizes.

Curve approximation and segmentation with the piecewise regression model

Once the model parameters have been estimated, a segmentation of the curve, equivalently represented by the vector of segment labels $\hat{\mathbf{z}} = (\hat{z}_1,\ldots,\hat{z}_m)$, where $\hat{z}_j \in \{1,\ldots,K\}$, can be derived by setting $\hat{z}_j = k$ if $j \in (\hat{\zeta}_k;\hat{\zeta}_{k+1}]$, the parameters $(\hat{\boldsymbol{\psi}},\hat{\boldsymbol{\zeta}})$ being the parameters provided by the dynamic programming procedure.

An approximation of the curve $\mathbf{y}$ is then given by the piecewise polynomial function $\hat{\mathbf{y}} = (\hat{y}_1,\ldots,\hat{y}_m)$ where $\hat{y}_j = \sum_{k=1}^{K}\hat{z}_{jk}\hat{\boldsymbol{\beta}}_k^T\mathbf{t}_j$, with $\hat{z}_{jk} = 1$ if $\hat{z}_j = k$ and $\hat{z}_{jk} = 0$ otherwise. The vectorial formulation of the approximated curve $\hat{\mathbf{y}}$ can be written as

$$\hat{\mathbf{y}} = \sum_{k=1}^{K}\hat{\mathbf{Z}}_k\mathbf{X}\hat{\boldsymbol{\beta}}_k, \tag{2.103}$$

where $\hat{\mathbf{Z}}_k$ is a diagonal matrix whose diagonal elements are $(\hat{z}_{1k},\ldots,\hat{z}_{mk})$.

However, it should be noticed that since the estimated curve (2.103) is computed as piecewise polynomial functions, the piecewise regression model does not necessarily guarantees a continuous smooth function approximation (since the binary indicator variables $\hat{z}_{ik}$ represent a hard segmentation of the curve). The piecewise approach is therefore more adapted for modeling curve presenting abrupt regime changes and may be less efficient for curve including regimes with smooth transitions.

2.6.5 Piecewise polynomial regression for a set of curves

This section provides an overview of the piecewise polynomial regression model in a context of modeling a set of curves and recalls the algorithm used for parameter estimation.

Modeling a set of curves by the piecewise polynomial regression model

Piecewise polynomial regression, generally used to model a single curve, as seen in section 2.6.4 (Brailovsky and Kempner, 1992; Chamroukhi et al., 2009b; Ferrari-Trecate and Muselli, 2002; McGee and Carleton, 1970), can also be used to model a set of curves (Chamroukhi et al., 2010; Hébrail et al., 2010; Hugueney et al., 2009). The parameters estimation is performed using dynamic programming (Bellman, 1961; Fisher, 1958; Lechevalier, 1990; Stone, 1961) due to the additivity of the optimized cost function over the K segments.

The piecewise polynomial regression model assumes that the curves $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ incorporate K polynomial regimes defined on K intervals whose bounds indexes can be denoted by $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_{K+1})$ with $\zeta_1 = 0$ and $\zeta_{K+1} = m$. This defines a partition of $\mathbf{Y}$ into K segments of curves of lengths $m_1, \ldots, m_K$ respectively:

$$\{y_{ij}|i \in [1, n] \text{ and } j \in I_1\}, \ldots, \{y_{ij}|i \in [1, n] \text{ and } j \in I_K\}$$

where $I_k = (\zeta_k, \zeta_{k+1}]$. Therefore, the piecewise polynomial regression model for the set of curves can be defined as follows:

$$\forall i = 1, \ldots, n,\ \forall j = 1, \ldots, m, \quad y_{ij} = \begin{cases} \boldsymbol{\beta}_1^T \mathbf{t}_j + \sigma_1 \epsilon_j & \text{if } j \in I_1 \\ \boldsymbol{\beta}_2^T \mathbf{t}_j + \sigma_2 \epsilon_j & \text{if } j \in I_2 \\ \vdots & \\ \boldsymbol{\beta}_K^T \mathbf{t}_j + \sigma_K \epsilon_j & \text{if } j \in I_K \end{cases}, \tag{2.104}$$

The model parameters denoted by $(\boldsymbol{\psi}, \boldsymbol{\zeta})$ where $\boldsymbol{\psi} = (\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \sigma_1^2, \ldots, \sigma_K^2)$ and $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_{K+1})$ are estimated by maximum likelihood.

Maximum likelihood estimation for the piecewise polynomial regression model

As in classical model-based learning problems where each observation is described by a feature vector (Hastie et al., 2010), we assume that the curves sample $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ is independent. Therefore, the distribution of a curve $\mathbf{y}_i$ is given by:

$$p(\mathbf{y}_i|\mathbf{t}; \boldsymbol{\psi}, \boldsymbol{\zeta}) = \prod_{k=1}^{K} \prod_{j \in I_k} \mathcal{N}\left(y_{ij}; \boldsymbol{\beta}_k^T \mathbf{t}_j, \sigma_k^2\right), \tag{2.105}$$

and the log-likelihood of the parameter vector $(\boldsymbol{\psi}, \boldsymbol{\zeta})$ characterizing the piecewise regression model, given the set of curves $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ is then written as follows:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\psi}, \boldsymbol{\zeta}) &= \log \prod_{i=1}^{n} p(\mathbf{y}_i|\mathbf{t}; \boldsymbol{\psi}, \boldsymbol{\zeta}) \\ &= \sum_{k=1}^{K} \sum_{i=1}^{n} \sum_{j \in I_k} \log \mathcal{N}\left(y_{ij}; \boldsymbol{\beta}_k^T \mathbf{t}_j, \sigma_k^2\right) \\ &= -\frac{1}{2} \sum_{k=1}^{K} \Big[\frac{1}{\sigma_k^2} \sum_{i=1}^{n} \sum_{j \in I_k} (y_{ij} - \boldsymbol{\beta}_k^T \mathbf{t}_j)^2 + n m_k \log \sigma_k^2\Big] + cst \end{aligned} \tag{2.106}$$

where m_k is the cardinal number of I_k. Maximizing this log-likelihood is equivalent to minimizing the criterion

$$J(\boldsymbol{\psi}, \boldsymbol{\zeta}) = \sum_{k=1}^{K} \Big[\frac{1}{\sigma_k^2} \sum_{i=1}^{n} \sum_{j \in I_k} \big(y_{ij} - \boldsymbol{\beta}_k^T \mathbf{t}_j\big)^2 + n m_k \log \sigma_k^2\Big] \tag{2.107}$$

with respect to $(\boldsymbol{\psi}, \boldsymbol{\zeta})$. The next section shows how the parameters $\boldsymbol{\psi}$ and $\boldsymbol{\zeta}$ can be estimated using dynamic programming.

Parameter estimation by dynamic programming

A dynamic programming procedure can be used to minimize the additive criterion (2.107) with respect to $(\boldsymbol{\psi}, \boldsymbol{\zeta})$ or equivalently to minimize (2.108) with respect to $\boldsymbol{\zeta}$:

$$\begin{aligned} C(\boldsymbol{\zeta}) &= \min_{\boldsymbol{\psi}} J(\boldsymbol{\psi}, \boldsymbol{\zeta}) = \sum_{k=1}^{K} \min_{(\boldsymbol{\beta}_k, \sigma_k^2)} \Big[\frac{1}{\sigma_k^2} \sum_{i=1}^{n} \sum_{j=\zeta_k+1}^{\zeta_{k+1}} \big(y_{ij} - \boldsymbol{\beta}_k^T \mathbf{t}_j\big)^2 + n m_k \log \sigma_k^2\Big] \\ &= \sum_{k=1}^{K} \Big[\frac{1}{\hat{\sigma}_k^2} \sum_{i=1}^{n} \sum_{j=\zeta_k+1}^{\zeta_{k+1}} (y_{ij} - \hat{\boldsymbol{\beta}}_k^T \mathbf{t}_j)^2 + n m_k \log \hat{\sigma}_k^2\Big], \end{aligned} \tag{2.108}$$

where $\hat{\boldsymbol{\beta}}_k$ and $\hat{\sigma}_k^2$ are given by:

$$\hat{\boldsymbol{\beta}}_k = \arg \min_{\boldsymbol{\beta}_k \in \mathbb{R}^{p+1}} \sum_{i=1}^{n} \sum_{j=\zeta_k+1}^{\zeta_{k+1}} (y_{ij} - \boldsymbol{\beta}_k^T \mathbf{t}_j)^2 = \Big[\sum_{i=1}^{n} \sum_{j=\zeta_k+1}^{\zeta_{k+1}} \mathbf{t}_j \mathbf{t}_j^T\Big]^{-1} \sum_{i=1}^{n} \sum_{j=\zeta_k+1}^{\zeta_{k+1}} y_{ij} \mathbf{t}_j \tag{2.109}$$

and

$$\hat{\sigma}_k^2 = \arg \min_{\sigma_k^2 \in \mathbb{R}^+} \frac{1}{\sigma_k^2} \sum_{i=1}^{n} \sum_{j=\zeta_k+1}^{\zeta_{k+1}} (y_{ij} - \hat{\boldsymbol{\beta}}_k^T \mathbf{t}_j)^2 + n m_k \log \sigma_k^2 = \frac{1}{n m_k} \sum_{i=1}^{n} \sum_{j=\zeta_k+1}^{\zeta_{k+1}} (y_{ij} - \hat{\boldsymbol{\beta}}_k^T \mathbf{t}_j)^2. \tag{2.110}$$

It can be seen that the criterion $C(\boldsymbol{\zeta})$ given by Equation (2.108) is additive over the K segments. Thanks to its additivity, this criterion can be optimized globally using a dynamic programming procedure (Bellman, 1961; Lechevalier, 1990; Stone, 1961). Dynamic programming considers that an optimal partition of the data into K segments is the union of an optimal partition into $K-1$ segments and one segment. Thus, by denoting by $C_1(a,b)$ the optimal cost within one segment whose elements indexes are $(a,b]$ with $0 \leq a < b \leq m$, the optimal costs $C_k(a,b)$ for a partition into k segments, $k = 2, \ldots, K$, is recursively computed as follows:

$$\begin{cases} C_1(a,b) = \min_{(\boldsymbol{\beta},\sigma^2)} \Big[\frac{1}{\sigma^2} \sum_{i=1}^{n} \sum_{j=a+1}^{b} \big(y_{ij} - \boldsymbol{\beta}^T \mathbf{t}_j\big)^2 + n(b-a) \log \sigma^2\Big] \\ \qquad = \frac{1}{\hat{\sigma}^2} \sum_{i=1}^{n} \sum_{j=a+1}^{b} (y_{ij} - \hat{\boldsymbol{\beta}}^T \mathbf{t}_j)^2 + n(b-a) \log \hat{\sigma}^2 & \text{(2.111a)} \\ C_k(a,b) = \min_{a \leq h \leq b} \left[C_{k-1}(a,h) + C_1(h+1,b)\right] \text{ for } k = 2, \ldots, K. & \text{(2.111b)} \end{cases}$$

where $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}^2$ are computed respectively according to the equations (2.109) and (2.110) by replacing $(\zeta_k, \zeta_{k+1}]$ by $(a, b]$, m_k by $(b-a)$ and $\hat{\boldsymbol{\beta}}_k$ by $\hat{\boldsymbol{\beta}}$. Thus, the algorithm works as follows:

Step 1. (Initialization) This step consists of computing the cost matrix $C_1(a, b)$ for one segment $(a, b]$ for $0 \leq a < b \leq m$ using (2.111a).

Step 2. (Dynamic programming procedure) This step consists of recursively computing the optimal cost $C_k(a, b)$ for $k = 2, \ldots, K$ and $0 \leq a < b \leq m$ using (2.111b).

Step 3. (Finding the optimal segmentation) The optimal segmentation can be deduced from the optimal costs $C_k(a, b)$. (For more details see appendix A of Brailovsky and Kempner (1992)).

This algorithm has a time complexity of $\mathcal{O}(Kp^2n^2m^2)$ which can be computationally expensive for large sample sizes.

Approximating a set of curves with the piecewise regression model

Once the model parameters are estimated, the set of curves can be approximated by a single "mean curve" $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_m)$ where each point from this curve is given by $\hat{y}_j = \sum_{k=1}^{K} \hat{z}_{jk} \hat{\boldsymbol{\beta}}_k^T \mathbf{t}_j$, $\forall\, j = 1, \ldots, m$ where $\hat{z}_{jk} = 1$ if $j \in (\hat{\zeta}_k, \hat{\zeta}_{k+1}]$ (i.e., the observation y_{ij} belongs to the kth regime) and $\hat{z}_{jk} = 0$ otherwise. The vectorial formulation of the "mean curve" $\hat{\mathbf{y}}$ can be written as

$$\hat{\mathbf{y}} = \sum_{k=1}^{K} \hat{\mathbf{Z}}_k \mathbf{X} \hat{\boldsymbol{\beta}}_k, \tag{2.112}$$

where $\hat{\mathbf{Z}}_k$ is a diagonal matrix whose diagonal elements are $(\hat{z}_{1k}, \ldots, \hat{z}_{mk})$, and $\mathbf{X}$ is the $m \times (p+1)$ regression matrix.

The piecewise regression model uses dynamic programming for parameter estimation. However, it is well-known that dynamic programming procedures are computationally expensive especially for large samples. In addition, the piecewise regression approach is more adapted for modeling curve presenting abrupt changes, since the "mean curve" is computed from a hard segmentation of the curves, and therefore may be less efficient for curves including smooth regime changes. For the case of a single curve, an alternative approach is to use a Hidden Markov Model Regression (Fridman, 1993) whose parameters are iteratively estimated by the Baum-Welch algorithm (Baum et al., 1970). This is presented in the next section.

2.6.6 Hidden Markov Model Regression

This section recalls the Hidden Markov Model Regression (Fridman, 1993) for curve modeling.

A general description of Hidden Markov Model Regression

In a Hidden Markov Model Regression (HMMR), the curve is assumed to be generated by the following regression model (Fridman, 1993):

$$y_j = \boldsymbol{\beta}_{z_j}^T \mathbf{t}_i + \sigma_{z_j} \epsilon_j \quad (j = 1, \ldots, m) \tag{2.113}$$

where z_j is a hidden discrete-valued variable taking its values in the set $\{1, \ldots, K\}$. The variable z_j controls the switching from one polynomial regression model to another of K models at each time t_j. The HMMR assumes that the hidden sequence $\mathbf{z} = (z_1, \ldots, z_n)$ is a homogeneous Markov chain of first order parametrized by the initial state distribution π and the transition matrix $\mathbf{A}$. The distribution of the latent sequence $\mathbf{z} = (z_1, \ldots, z_n)$ is defined by Equation (2.64).

Note that if the curve we aim to model consists of successive contiguous regimes, we can impose the following constraints on the transition probabilities. These constraints consist of setting $p(z_j = k | z_{j-1} = \ell) = 0$ if $k < \ell$ or if $k > \ell + 1$ which imply that no transitions are allowed for the states whose indexes are lower than the current state and no jumps of more than one state are possible. This leads to a particular case of the well known left-right model (Rabiner, 1989) (see section 2.5.3).

Parameter estimation

From the model defined by Equation (2.113), it can be proved that, conditionally on a regression model k ($z_j = k$), y_j has a Gaussian distribution with mean $\boldsymbol{\beta}_k^T \mathbf{t}_j$ and variance σ_k^2. Thus, the HMMR is parameterized by the parameter vector $\boldsymbol{\Psi} = (\pi, \mathbf{A}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \sigma_1^2, \ldots, \sigma_K^2)$. The parameter vector $\boldsymbol{\Psi}$ is estimated by the maximum likelihood method. The log-likelihood to be maximized in this case is written as

$$\begin{aligned}\mathcal{L}(\boldsymbol{\Psi}; \mathbf{y}) &= \log p(\mathbf{y}; \boldsymbol{\Psi}) \\ &= \log \sum_{z_1} \cdots \sum_{z_m} p(z_1; \pi) \prod_{j=2}^{m} p(z_j | z_{j-1}; \mathbf{A}) \prod_{j=1}^{m} \mathcal{N}(y_j; \boldsymbol{\beta}_{z_j}^T \mathbf{t}_j, \sigma_{z_j}^2).\end{aligned} \tag{2.114}$$

Since this log-likelihood can not be maximized in a closed form, this is done by the EM algorithm (Dempster et al., 1977), which is known as the Baum-Welch algorithm (Baum et al., 1970) in the context of HMMs.

The time complexity of both the Forward and Backward procedure of standard HMMs for unidimensional sequence observation is of $\mathcal{O}(K^2 m)$ per iteration (Cappé et al., 2005; Murphy, 2002; Rabiner and Juang, 1993). In addition, in this regression context, the calculation of the regression coefficients in the M-step requires an inversion of $(p+1) \times (p+1)$ matrix and a multiplication by the observation sequence of length m which is done with a complexity of $\mathcal{O}(p^2 m)$ (see for example (Makarenkov and Legendre, 1999)). The Baum-Welch algorithm has therefore a time complexity of $\mathcal{O}(I_{\text{EM}} K^2 p^2 m)$, where I_{EM} is the number of iterations of the EM (Baum-Welch) algorithm.

Curve approximation and segmentation with Hidden Markov Model Regression

To approximate the curve $\mathbf{y}$ at each time t_j, a common way is to combine the different regression models using the posterior component probabilities (the smoothing probabilities) given the estimated model parameters $\hat{\boldsymbol{\Psi}}$ which we denote by $\hat{\tau}_{jk} = p(z_j = k|\mathbf{y}; \hat{\boldsymbol{\Psi}})$ for the kth regression model. These probabilities are computed by using the so-called forward-backward procedure (Rabiner, 1989) (see Equation (2.75) in section 2.5.3 and Appendix A.4 for the details). Thus, each point $\hat{y}_j$ of the estimated curve is given by:

$$\hat{y}_j = \sum_{k=1}^{K} \hat{\tau}_{jk} \hat{\boldsymbol{\beta}}_k^T \mathbf{t}_j, \quad (j = 1, \ldots, m). \tag{2.115}$$

The vectorial formulation of the approximated curve $\hat{\mathbf{y}}$ can be written as

$$\hat{\mathbf{y}} = \sum_{k=1}^{K} \hat{\mathbf{W}}_k \mathbf{X} \hat{\boldsymbol{\beta}}_k, \tag{2.116}$$

where $\hat{\mathbf{W}}_k$ is a diagonal matrix whose diagonal elements are $(\hat{\tau}_{1k}, \ldots, \hat{\tau}_{mk})$, and $\mathbf{X}$ is the $m \times (p+1)$ regression matrix.

Moreover, given the estimated model parameters $\hat{\boldsymbol{\Psi}}$, a segmentation of the curve can be deduced by computing the optimal sequence $\hat{\mathbf{z}} = (\hat{z}_1, \ldots, \hat{z}_m)$ of the segment labels by using the Viterbi algorithm (Forney, 1973; Viterbi, 1967) (c.f., Appendix A.6).

It is worth to note that, while HMMs can be applied to model sets of curves and we would have a single HMM for each set of curves, as we will see it in section 2.7.4, to my knowledge, there is no an existing method for approximating a set of curves by a single curve for HMM regression. As alternative, one can think about an empirical mean of the n smoothed curves to give a single representation for a set of curves.

Illustration

Figure 2.8 shows fitted curves to noisy nonlinear regression function using different models.

2.7 Curve clustering

In this section, we describe some models and algorithms that address the curve clustering problem. The clustering methods presented here are model-based curve clustering approaches relying on regression mixtures, including polynomial regression, spline regression and piecewise regression. In this context, a generative model has been developed by Gaffney (2004); Gaffney and Smyth (2004) which consists of clustering curves with polynomial regression mixtures. Another approach based on splines is concerned with clustering sparsely sampled curves (James and Sugar, 2003). More recently, Liu

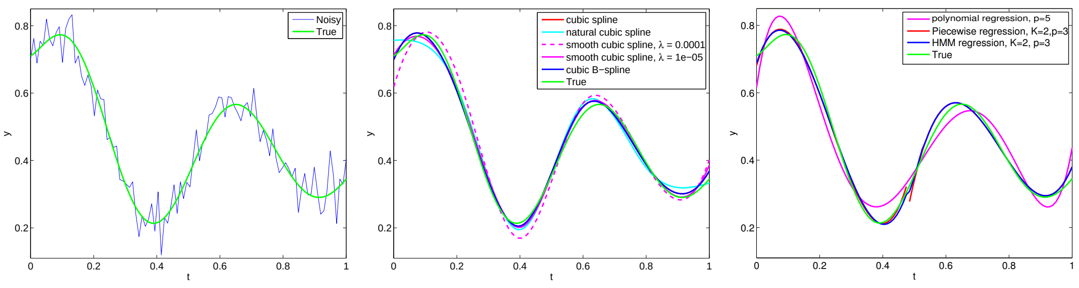


Figure 2.8: Plot of fitted curves to the noisy regression function given in the left plot and given by $y = 0.41 + 0.3\exp(-t)(\sin(3.8\pi t) + \exp(-9t)) + \epsilon_t$; $\epsilon_t \sim \mathcal{N}(0, 0.05^2)$; for $m = 100$ data points. The plot in the middle shows the solutions obtained with different polynomial spline regression models. All splines are of order $M = 4$ (cubic) with internal knots placement at [0.2 0.4 0.6 0.8]. The right plot shows the fitted curves obtained with a polynomial fit using a polynomial of degree $p = 3$, polynomial piecewise regression and polynomial HMM regression with $K = 2$ polynomial components of degree $p = 3$.

and Yang (2009) proposed a curve clustering approach using B-spline basis functions to model each cluster. All these approaches use the EM algorithm to estimate the model parameters.

A distance-based curve clustering approach, that allows for fitting several constant (or polynomial) models to the curves, consists of the piecewise regression model using a K-means-like algorithm (Hébrail et al., 2010; Hugueney et al., 2009). This approach simultaneously performs curve clustering and optimal segmentation using dynamic programming.

In this section we describe these generative approaches for curve clustering. Next we consider the piecewise regression approach for curve clustering of Hébrail et al. (2010) from a probabilistic prospective, and we give a model-based curve clustering which is the piecewise regression mixture. The proposed approach which uses the EM algorithm in the case of maximum likelihood estimation and the CEM algorithm in the case of classification likelihood estimation. It also performs both curve clustering and optimal segmentation using dynamic programming.

2.7.1 Polynomial regression mixture and spline regression mixture

The mixture model

In this section we describe curve clustering approaches based on polynomial regression mixture models (PRM) and polynomial spline regression mixtures (PSRM) (Gaffney, 2004). These approaches assume that each curve is drawn from one of R clusters of curves which are mixed at random in proportion to the relative cluster sizes $(\alpha_1, \ldots, \alpha_R)$. Each cluster of curves is supposed to be a set of homogeneous curves modeled by either a polynomial regression model or a spline regression model. Thus, the mixture density of a curve $\mathbf{y}_i$ $(i = 1, \ldots, n)$ can be written as:

$$p(\mathbf{y}_i|\mathbf{t};\boldsymbol{\Psi}) = \sum_{r=1}^{R} \alpha_r \prod_{j=1}^{m} \mathcal{N}(y_{ij};\boldsymbol{\beta}_r^T\mathbf{t}_j, \sigma_r^2) = \sum_{r=1}^{R} \alpha_r\ \mathcal{N}(\mathbf{y}_i;\mathbf{X}\boldsymbol{\beta}_r, \sigma_r^2\mathbf{I}_m), \tag{2.117}$$

where $\mathbf{X}$ is the regression matrix, $\boldsymbol{\beta}_r$ is the polynomial coefficient vector of the cluster r $(r = 1, \ldots, R)$, the α_r's are the non-negative mixing proportions that sum to 1 and $\boldsymbol{\Psi} = (\alpha_1, \ldots, \alpha_r, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_R)$ with $\boldsymbol{\theta}_r = (\boldsymbol{\beta}_r, \sigma_r^2)$, σ_r^2 being the noise variance for the cluster r. The unknown parameter vector $\boldsymbol{\Psi}$ is estimated by maximum likelihood via the EM algorithm (Gaffney, 2004).

Parameter estimation via the EM algorithm

Given an i.i.d training set of n curves $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ regularly sample at the time points $\mathbf{t} = (t_1, \ldots, t_m)$, the log-likelihood of $\boldsymbol{\Psi}$ is given by:

$$\mathcal{L}(\boldsymbol{\Psi};\mathbf{Y},\mathbf{t}) = \log \prod_{i=1}^{n} p(\mathbf{y}_i;\boldsymbol{\Psi}) = \sum_{i=1}^{n} \log \sum_{r=1}^{R} \alpha_r\ \mathcal{N}(\mathbf{y}_i;\mathbf{X}\boldsymbol{\beta}_r, \sigma_r^2\mathbf{I}_m). \tag{2.118}$$

The log-likelihood is maximized by the EM algorithm. Before giving the EM steps, the complete-data log-likelihood is given by:

$$\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{h}) \quad = \quad \sum_{i=1}^{n}\sum_{r=1}^{R} h_{ir} \log \alpha_r + \sum_{i=1}^{n}\sum_{r=1}^{R} h_{ir} \log \mathcal{N}(\mathbf{y}_i;\boldsymbol{\beta}_r^T\mathbf{t}_j, \sigma_r^2\mathbf{I}_m) \tag{2.119}$$

where $\mathbf{h} = (h_1, \ldots, h_n)$ is the vector of cluster labels for the n curves and h_{ir} is an indicator binary-valued variable such that $h_{ir} = 1$ if $h_i = r$ (i.e., if $\mathbf{y}_i$ is generated by the cluster r). The EM algorithm for PRMs and PSRMs starts with an initial model parameters $\boldsymbol{\Psi}^{(0)}$ and alternates between the two following steps until convergence:

E-step: Compute the expected complete-data log-likelihood given the curves $\mathbf{Y}$, the time vector $\mathbf{t}$ and the current value of the parameter $\boldsymbol{\Psi}$ denoted by $\boldsymbol{\Psi}^{(q)}$:

$$\begin{aligned} Q(\boldsymbol{\Psi},\boldsymbol{\Psi}^{(q)}) &= \mathbb{E}\big[\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{h})|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)}\big] \\ &= \sum_{i=1}^{n}\sum_{r=1}^{R} \mathbb{E}\big[h_{ir}|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)}\big] \log \alpha_r + \sum_{i=1}^{n}\sum_{r=1}^{R} \mathbb{E}\big[h_{ir}|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)}\big] \log \mathcal{N}(\mathbf{y}_i;\mathbf{X}\boldsymbol{\beta}_r, \sigma_r^2\mathbf{I}_m) \\ &= \sum_{i=1}^{n}\sum_{r=1}^{R} \tau_{ir}^{(q)} \log \alpha_r + \sum_{i=1}^{n}\sum_{r=1}^{R} \tau_{ir}^{(q)} \log \mathcal{N}(\mathbf{y}_i;\mathbf{X}\boldsymbol{\beta}_r, \sigma_r^2\mathbf{I}_m) \end{aligned} \tag{2.120}$$

where

$$\tau_{ir}^{(q)} = p(h_i = r|\mathbf{y}_i,\mathbf{t};\boldsymbol{\Psi}^{(q)}) = \frac{\alpha_r^{(q)}\mathcal{N}\big(\mathbf{y}_i;\mathbf{X}\boldsymbol{\beta}_r^{T(q)}, \sigma_r^{2(q)}\mathbf{I}_m\big)}{\sum_{r'=1}^{R}\alpha_{r'}^{(q)}\mathcal{N}(\mathbf{y}_i;\mathbf{X}\boldsymbol{\beta}_{r'}^{(q)T}, \sigma_{r'}^{2(q)}\mathbf{I}_m)} \tag{2.121}$$

is the posterior probability that the curve $\mathbf{y}_i$ is generated by the cluster r. This step therefore only requires the computation of the posterior cluster probabilities $\tau_{ir}^{(q)}$ $(i = 1, \ldots, n)$ for each of the R clusters.

M-step: Compute the update $\mathbf{\Psi}^{(q+1)}$ fo $\mathbf{\Psi}$ by maximizing the Q-function (2.120) with respect to $\mathbf{\Psi}$. The two terms of the Q-function are maximized separately. The first term, that is the function $\sum_{i=1}^{n}\sum_{r=1}^{R}\tau_{ir}^{(q)}\log\alpha_r$ is maximized with respect to $(\alpha_1,\ldots,\alpha_R)$ subject to the constraint $\sum_{r=1}^{R}\alpha_r = 1$ using Lagrange multipliers which gives the following updates:

$$\alpha_r^{(q+1)} = \frac{1}{n}\sum_{i=1}^{n}\tau_{ir}^{(q)} \quad (r=1,\ldots,R). \tag{2.122}$$

The second term of (2.120) can also be decomposed independently as a sum of R functions of $(\boldsymbol{\beta}_r,\sigma_r^2)$ to perform R separate maximizations. The maximization of each of the R functions, that is $\sum_{i=1}^{n}\tau_{ir}^{(q)}\log\mathcal{N}(\mathbf{y}_i;\mathbf{X}\boldsymbol{\beta}_r,\sigma_r^2\mathbf{I}_m)$, corresponds therefore to solving a weighted least-squares problem. The solution of this problem is straightforward and is given by:

$$\boldsymbol{\beta}_r^{(q+1)} = \left(\mathbf{X}^{*T}\mathbf{W}_r^{(q)}\mathbf{X}^*\right)^{-1}\mathbf{X}^{*T}\mathbf{W}_r^{(q)}\mathbf{y}^* \tag{2.123}$$

$$\sigma_r^{2(q+1)} = \frac{1}{\sum_{i=1}^{n}\tau_{ir}^{(q)}}(\mathbf{y}^*-\mathbf{X}^{*T}\boldsymbol{\beta}_r^{(q+1)})^T\mathbf{W}_r^{(q)}(\mathbf{y}^*-\mathbf{X}^{*T}\boldsymbol{\beta}_r^{(q+1)}) \tag{2.124}$$

where $\mathbf{X}^*$ is a matrix composed of n copies of the regression matrix $\mathbf{X}$ (including spline regression and B-spline regression) such that we have $\mathbf{X}^* = (\mathbf{X}^T,\ldots,\mathbf{X}^T)^T$, the vector $\mathbf{y}^*$ is an $nm\times 1$ vector composed of the n curves by stacking them one curve after another, that is $\mathbf{y}^* = (\mathbf{y}_1^T,\ldots,\mathbf{y}_n^T)^T$ and $\mathbf{W}_r^{(q)}$ is the $nm\times nm$ diagonal matrix whose diagonal elements are $(\underbrace{\tau_{1r}^{(q)},\ldots,\tau_{1r}^{(q)}}_{m\text{ times}},\ldots,\underbrace{\tau_{nr}^{(q)},\ldots,\tau_{nr}^{(q)}}_{m\text{ times}})$.

2.7.2 Piecewise regression for curve clustering via a K-means-like algorithm

In this section we recall the piecewise polynomial regression model proposed by Hébrail et al. (2010); Hugueney et al. (2009) for clustering and optimal segmentation of a set of curves. In this approach, the authors use a piecewise constant regression model for both clustering and segmenting a set of curves. An euclidean distance criterion is used to perform the segmentation of each cluster of curves. The segmentation is performed in an optimal way using dynamic programming thanks to the additivity of the distance criterion over the set of segments for each cluster. The euclidean distance criterion, relative to each cluster of curves, is integrated in an overall sum of euclidean distances over the clusters and is iteratively minimized to find a partition of the curves into R clusters. The resulting algorithm is a K-means-like algorithm. More precisely, the clustering and segmentation algorithm proposed in (Hébrail et al., 2010; Hugueney et al., 2009) optimizes the following distance criterion:

$$E\left(\mathbf{h},\{I_{rk}\},\{\mu_{rk}\}\right) = \sum_{r=1}^{R}\sum_{i|h_i=r}\sum_{k=1}^{K_r}\sum_{j\in I_{rk}}(y_{ij}-\mu_{rk})^2 \tag{2.125}$$

where $I_{rk} = (\zeta_{rk}, \zeta_{r,k+1}]$ represent the indexes of elements of segment k ($k = 1, \ldots, K_r$) for cluster r and μ_{rk} its constant mean, K_r being its own total number of segments. It can be seen, as noted by Hébrail et al. (2010), that the distance criterion (2.125) can be rewritten as

$$E\left(\mathbf{h}, \{I_{rk}\}, \{\mu_{rk}\}\right) = \sum_{r=1}^{R} \sum_{i|h_i=r} \|\mathbf{y}_i - \mathbf{g}_r\|^2 \tag{2.126}$$

where $\mathbf{g}_r = (g_{r1}, \ldots, g_{rm})$ is an $m \times 1$ dimensional vector such that $g_{rj} = \mu_{rk}$ if $j \in I_{rk}$ for all $j = 1, \ldots, m$ (i.e., the jth observation y_{ij} belongs to segment k of cluster r). Thus, the vector of piecewise mean constants $\mathbf{g}_r$ can be seen as the "centroid" of cluster r ($r = 1, \ldots, R$) and the criterion (2.126) as the distortion criterion optimized by the standard K-means in the case of multidimensional data (c.f., Equation (2.26) in section 2.3.1). This criterion is then iteratively optimized as follows. After starting with an initial cluster partition $\mathbf{h}^{(0)}$ (e.g., initialized randomly), the K-means-like algorithm alternating between the two following steps until convergence.

Relocation step: This step consists of finding the optimal piecewise constant prototype for a given cluster r as follows:

- Find the segmentation of each cluster r by minimizing the additive criterion relative to each cluster r given by (c.f., Equation (2.125)):
$E_r(\mathbf{h}^{(q)}, \{I_{rk}\}, \{\mu_{rk}\}) = \sum_{k=1}^{K_r} \sum_{i|h_i^{(q)}=r} \sum_{j \in I_{rk}} (y_{ij} - \mu_{rk})^2$

  w.r.t the segment bounds $\{I_{rk}\}$ and the constant means $\{\mu_{rk}\}$ for each segment. Due to its additivity over the segments, the segmentation can therefore be performed in an optimal way using dynamic programming, similarly as in section 2.6.5 by using the schemes (2.111a) and (2.111b) for which the constant means μ_{rk} are computed as the arithmetic mean:
$\mu_{rk} = \frac{1}{\#I_{rk} \sum_{i|h_i^{(q)}=r}} \sum_{i|h_i^{(q)}=r} \sum_{j \in I_{rk}} y_{ij}$.

- Each cluster representative is relocated to the piecewise constant prototype, which is the arithmetic mean of all data points assigned to it, that is, for each cluster r find a piecewise constant prototype $\mathbf{g}_k^{(q)}$ with elements $\mu_{rk}^{(q)} = \frac{1}{\#I_{rk}^{(q)} \sum_{i|h_i^{(q)}=r}} \sum_{i|h_i^{(q)}=r} \sum_{j \in I_{rk}^{(q)}} y_{ij}$ where $\{I_{rk}^{(q)}\}$ corresponds to the optimal segmentation of cluster r provided by dynamic programming.

Assignment step: Assign each curve $\mathbf{y}_i$ to the near piecewise constant prototype in the sense of the euclidean distance: $h_i^{(q+1)} = \arg\min_{1 \leq r \leq R} \|\mathbf{y}_i - \mathbf{g}_r^{(q)}\|^2$.

Notice that this approach can be easily generalized to perform piecewise polynomial regression in which the cluster representative is a piecewise polynomial function rather than than a piecewise constant function. Additional details can be found in Hébrail et al. (2010).

2.7.3 Piecewise polynomial regression mixture

In this section, we will use the piecewise polynomial regression model in a mixture model-based curve clustering framework. Each curve $\mathbf{y}_i$ $(i = 1, \ldots, n)$ is assumed to be generated by a piecewise regression model among R models defined by (2.105), with a prior probability α_r. Thus the distribution of a curve is given by the following piecewise polynomial regression mixture model:

$$p(\mathbf{y}_i|\mathbf{t};\boldsymbol{\Psi}) = \sum_{r=1}^{R} \alpha_r \prod_{k=1}^{K_r} \prod_{j\in I_{rk}} \mathcal{N}(y_{ij};\boldsymbol{\beta}_{rk}^T\mathbf{t}_j, \sigma_{rk}^2), \quad (2.127)$$

where $\in I_{rk}$ is the set of indexes of elements of segment k for the cluster r $(r = 1, \ldots, R)$, $\boldsymbol{\beta}_{rk}$ is the $p+1$-dimensional vector of polynomial coefficients and α_r $(r = 1, \ldots, R)$ are the non-negative mixing proportions that sum to 1. The model parameters $\boldsymbol{\Psi}$ can therefore denoted by:

$$\boldsymbol{\Psi} = (\alpha_1, \ldots, \alpha_R, \boldsymbol{\psi}_1, \ldots, \boldsymbol{\psi}_R, \boldsymbol{\zeta}_1, \ldots, \boldsymbol{\zeta}_R)$$

where $\boldsymbol{\psi}_r = (\boldsymbol{\beta}_{r1}, \ldots, \boldsymbol{\beta}_{rK_r}, \sigma_{r1}^2, \ldots, \sigma_{rK_r}^2)$ and $\boldsymbol{\zeta}_r = (\zeta_{r1}, \ldots, \zeta_{r,K_r+1})$ are respectively the set of polynomial coefficients and noise variances, and the set of transition points which correspond to the segmentation of the cluster r $(r = 1, \ldots, R)$.

Maximum likelihood estimation via EM

Parameter estimation can be performed in maximum likelihood framework. Assume we have the set of i.i.d n curves $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ sampled at the time points $\mathbf{t}$. The log-likelihood of $\boldsymbol{\Psi}$ for the observed data is therefore written as

$$\mathcal{L}(\boldsymbol{\Psi};\mathbf{Y},\mathbf{t}) = \log \prod_{i=1}^{n} p(\mathbf{y}_i|\mathbf{t};\boldsymbol{\Psi}) = \sum_{i=1}^{n} \log \sum_{r=1}^{R} \alpha_r \prod_{k=1}^{K_r} \prod_{j\in I_{rk}} \mathcal{N}\left(y_{ij};\boldsymbol{\beta}_{rk}^T\mathbf{t}_j, \sigma_{rk}^2\right). \quad (2.128)$$

The maximization of this log-likelihood can be performed by the EM algorithm. In this framework, the complete-data log-likelihood, for a particular configuration $\mathbf{h} = (h_1, \ldots, h_n)$, where h_i is the cluster label of the ith curve, is given by:

$$\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{t},\mathbf{h}) = \sum_{i=1}^{n}\sum_{r=1}^{R} h_{ir} \log \alpha_r + \sum_{i=1}^{n}\sum_{r=1}^{R}\sum_{k=1}^{K_r}\sum_{j\in I_{rk}} h_{ir} \log \mathcal{N}(y_{ij};\boldsymbol{\beta}_{rk}^T\mathbf{t}_j, \sigma_{rk}^2) \quad (2.129)$$

where h_{ir} is an indicator binary-valued variable such that $h_{ir} = 1$ if $\mathbf{y}_i$ is generated by the cluster r. The next paragraph shows how the observed-data log-likelihood is maximized by the EM algorithm to perform curve clustering and segmentation.

The EM algorithm for piecewise polynomial regression mixture

The EM algorithm for the polynomial piecewise regression mixture model starts with an initial solution $\boldsymbol{\Psi}^{(0)}$ (e.g., computed from a random partition) and alternates between the two following steps until convergence (e.g., when there is no longer change in the relative variation of the log-likelihood):

E-step Compute the expected complete-data log-likelihood given the curves $\mathbf{Y}$, the time vector $\mathbf{t}$ and the current value of the parameters denoted by $\boldsymbol{\Psi}^{(q)}$:

$$\begin{aligned} Q(\boldsymbol{\Psi}, \boldsymbol{\Psi}^{(q)}) &= \mathbb{E}\big[\mathcal{L}_c(\boldsymbol{\Psi}; \mathbf{Y}, \mathbf{t}, \mathbf{h})|\mathbf{Y}, \mathbf{t}; \boldsymbol{\Psi}^{(q)}\big] \\ &= \sum_{i=1}^{n}\sum_{r=1}^{R}\mathbb{E}\big[h_{ir}|\mathbf{Y}, \mathbf{t}; \boldsymbol{\Psi}^{(q)}\big]\log\alpha_r \\ &\quad + \sum_{i=1}^{n}\sum_{r=1}^{R}\sum_{k=1}^{K_r}\sum_{j\in I_{rk}^{(q)}}\mathbb{E}\big[h_{ir}|\mathbf{Y}, \mathbf{t}; \boldsymbol{\Psi}^{(q)}\big]\log\mathcal{N}(y_{ij}; \boldsymbol{\beta}_{rk}^T\mathbf{t}_j, \sigma_{rk}^2) \\ &= \sum_{i=1}^{n}\sum_{r=1}^{R}\tau_{ir}^{(q)}\log\alpha_r + \sum_{i=1}^{n}\sum_{r=1}^{R}\sum_{k=1}^{K_r}\sum_{j\in I_{rk}^{(q)}}\tau_{ir}^{(q)}\log\mathcal{N}(y_{ij}; \boldsymbol{\beta}_{rk}^T\mathbf{t}_j, \sigma_{rk}^2) \quad (2.130)\end{aligned}$$

where

$$\begin{aligned} \tau_{ir}^{(q)} &= p(h_{ir} = 1|\mathbf{y}_i, \mathbf{t}; \boldsymbol{\Psi}^{(q)}) = p(h_i = r|\mathbf{y}_i, \mathbf{t}; \boldsymbol{\Psi}^{(q)}) \\ &= \frac{\alpha_r^{(q)}\prod_{k=1}^{K_r}\prod_{j\in I_{rk}^{(q)}}\mathcal{N}\big(y_{ij}; \boldsymbol{\beta}_{rk}^{T(q)}\mathbf{t}_j, \sigma_{rk}^{2(q)}\big)}{\sum_{r'=1}^{R}\alpha_{r'}^{(q)}\prod_{k=1}^{K_r}\prod_{j\in I_{r'k}^{(q)}}\mathcal{N}(y_{ij}; \boldsymbol{\beta}_{r'k}^{(q)T}\mathbf{t}_j, \sigma_{r'k}^{2(q)})} \end{aligned} \quad (2.131)$$

is the posterior probability that the curve $\mathbf{y}_i$ belongs to the cluster r. This step therefore only requires the computation of the posterior cluster probabilities $\tau_{ir}^{(q)}$ $(i = 1, \ldots, n)$ for each of the R clusters.

M-step Compute the parameter update $\boldsymbol{\Psi}^{(q+1)}$ by maximizing the Q-function (2.130) with respect to $\boldsymbol{\Psi}$. The maximization of the Q-function is performed by separately maximizing the two terms of (2.130) as follows. The first term, that is the function $\sum_{i=1}^{n}\sum_{r=1}^{R}\tau_{ir}^{(q)}\log\alpha_r$, is maximized with respect to $(\alpha_1, \ldots, \alpha_R) \in [0, 1]^R$ subject to the constraint $\sum_{r=1}^{R}\alpha_r = 1$ using Lagrange multipliers, which gives the following updates:

$$\alpha_r^{(q+1)} = \frac{1}{n}\sum_{i=1}^{n}\tau_{ir}^{(q)} \quad (r = 1, \ldots, R). \quad (2.132)$$

To find the regression parameters and the segmentation $(\boldsymbol{\psi}_r, \boldsymbol{\zeta}_r)$ for each of the R clusters (note that here the segmentation of cluster r is defined via the parameters $\boldsymbol{\zeta}_r$), it can be seen that the second term of (2.129) can be decomposed independently into R independent functions, each of the R functions is given as in (2.106). Therefore, this consists of performing R separate maximizations, or equivalently minimizations, w.r.t $(\boldsymbol{\psi}_r, \boldsymbol{\zeta}_r)$ where each of these minimizations involves a fuzzy cluster r $(r = 1, \ldots, R)$ and which we perform by dynamic programming thanks to the additivity of the resulting function over the segments. The updating rules for the regression parameters for each cluster are given similarly as in (2.109) and (2.110).

After convergence of the EM algorithm, a partition of the curves can then deduced by assigning each curve to the class maximizing the posterior cluster probabilities

(2.131), that is:

$$\hat{h}_i = \arg\max_{1\leq r\leq R} \tau_{ir}(\hat{\boldsymbol{\Psi}}), \quad (i = 1, \ldots, n). \tag{2.133}$$

The CEM algorithm for piecewise polynomial regression mixture

In this paragraph we adopt the classification version of EM, that is the CEM algorithm (Celeux and Govaert, 1992) to optimize the polynomial piecewise regression mixture model for curve clustering. The CEM algorithm maximizes the complete-data log-likelihood (2.129) w.r.t the model parameters $\boldsymbol{\Psi}$ and the partition represented by the vector cluster labels $\mathbf{h}$ in an iterative manner in the following way. After starting with an initial model parameters $\boldsymbol{\Psi}^{(0)}$ (e.g., computed from a randomly chosen partition), the CEM algorithm alternates between the two following steps at each iteration q until convergence (e.g., when the is no longer change in the partition or in the relative variation of the complete-data log-likelihood):

Step 1: Update the cluster labels for the current model defined by $\boldsymbol{\Psi}^{(q)}$ by maximizing the complete-data log-likelihood (2.129) w.r.t to the vector $\mathbf{h}$

$$\mathbf{h}^{(q+1)} = \arg\max_{\mathbf{h}\in\{1,\ldots,R\}^n} \mathcal{L}_c(\mathbf{h}, \boldsymbol{\Psi}^{(q)}). \tag{2.134}$$

Step 2: Update the model parameters for the current partition defined by $\mathbf{h}^{(q)}$ by maximizing the complete-data log-likelihood w.r.t to the model parameters $\boldsymbol{\Psi}$:

$$\boldsymbol{\Psi}^{(q+1)} = \arg\max_{\boldsymbol{\Psi}\in\boldsymbol{\Omega}} \mathcal{L}_c(\mathbf{h}^{(q+1)}, \boldsymbol{\Psi}). \tag{2.135}$$

This step consists of estimating a piecewise polynomial regression model for the set of curves for each of the R clusters. This is performed using a dynamic programming procedure (see sections 2.6.5 and 2.6.5).

This scheme is equivalent to integrating a classification step between the E-step and the M-step of the EM algorithm (Celeux and Govaert, 1992):

E-step: Compute the posterior cluster probabilities $\tau_{ir}^{(q)}$ $(i = 1, \ldots, n)$ given by Equation (2.131) for each of the R clusters.

C-step: Compute a hard partition of the n curves by estimating the cluster labels through the MAP rule:

$$h_i^{(q)} = \arg\max_{1\leq r\leq R} \tau_{ir}^{(q)} \ (i = 1, \ldots, n). \tag{2.136}$$

M-step: Update the model parameters by computing the parameter $\boldsymbol{\Psi}^{(q+1)}$ which maximizes the complete-data log-likelihood (2.129) with respect to $\boldsymbol{\Psi}$ given the estimated cluster labels $\mathbf{h}^{(q)}$. The mixing proportions α_r's are updated by maximizing the function $\sum_{i=1}^{n}\sum_{r=1}^{R} h_{ir}^{(q)} \log \alpha_r$ w.r.t $(\alpha_1, \ldots, \alpha_R) \in [0,1]^R$ subject to the constraint $\sum_{r=1}^{R} \alpha_r = 1$ using Lagrange multipliers which gives the following updates:

$$\alpha_r^{(q+1)} = \frac{1}{n}\sum_{i=1}^{n} h_{ir}^{(q)} \quad (r = 1, \ldots, R). \tag{2.137}$$

The regression parameters and the segmentation $(\boldsymbol{\psi}_r, \boldsymbol{\zeta}_r)$ for each of the R clusters are updated by maximizing the function (see the second term in Equation (2.129), that is:

$$\mathcal{J}\big(\mathbf{h}^{(q)}, \{\boldsymbol{\beta}_{rk}, \sigma_{rk}^2, I_{rk}\}\big) = \sum_{r=1}^{R} \mathcal{J}_r\big(\mathbf{h}^{(q)}, \{\boldsymbol{\beta}_{rk}, \sigma_{rk}^2, I_{rk}\}_{k=1}^{K_r}\big) \tag{2.138}$$

with

$$\begin{aligned}\mathcal{J}_r\big(\mathbf{h}^{(q)}, \{\boldsymbol{\beta}_{rk}, \sigma_{rk}^2, I_{rk}\}_{k=1}^{K_r}\big) &= \sum_{k=1}^{K_r} \sum_{i|h_i^{(q)}=r} \sum_{j \in I_{rk}} \log \mathcal{N}(y_{ij}; \boldsymbol{\beta}_{rk}^T \mathbf{t}_j, \sigma_{rk}^2) \\ &= \sum_{k=1}^{K_r} \Big[\sum_{i|h_i^{(q)}=r} \sum_{j \in I_{rk}} \big(\frac{y_{ij} - \boldsymbol{\beta}_{rk}^T \mathbf{t}_j}{\sigma_{rk}}\big)^2 + n_r^{(q)} m_{rk} \log \sigma_{rk}^2 \Big] + cst\end{aligned} \tag{2.139}$$

for $r = 1, \ldots, R$ where I_{rk} is the cardinal number of I_{rk} and $n_r^{(q)}$ is the number of curved assigned to the cluster r at the current iteration q.

Thus, we perform R independent maximizations by decomposing (2.138) into R functions, each of them is given by (2.139) and is relative to the cluster r $(r = 1, \ldots, R)$. The criterion (2.139) is itself additive on k and therefore can be optimized in an optimal way by using dynamic programming. The optimized function for each set of curves belonging to the cluster r is close to the criterion (2.107) for which the optimization procedure and the estimated parameters have been provided (c.f., section 2.6.5).

It is therefore worth mentioning that the complete-data log-likelihood (2.129) optimized by the CEM algorithm, as it can be deduced from Equations (2.138, 2.139), is equivalent to the criterion optimized by the K-means-like algorithm of Hébrail et al. (2010) (c.f., Equation (2.125)) in the case of polynomial piecewise regression, if particular constraints are imposed on the mixing proportions α_r's and the variances σ_{rk}^2's: $\alpha_r = \frac{1}{R}$ $\forall R$ and $\sigma_{rk}^2 = \sigma_r^2$ $\forall k = 1, \ldots, K_r$ for each cluster r. Therefore, the CEM algorithm for piecewise polynomial regression mixture is a probabilistic view for hard curve clustering with the K-means-like algorithm.

2.7.4 Curve clustering with Hidden Markov Models

Hidden Markov Models can also be integrated in a mixture framework to perform sequence clustering, as developed by Smyth (1996). In this probabilistic model-based clustering (Alon et al., 2003; Smyth, 1996), an observation sequence (in this case a curve) is assumed to be generated according to a mixture distribution of R components, each of the R components is an HMM. Formally, the mixture distribution of each curve $\mathbf{y}_i$ $(i = 1, \ldots, n)$ is defined as

$$p(\mathbf{y}_i|\mathbf{t}; \boldsymbol{\Psi}) = \sum_{r=1}^{R} \alpha_r \; p(\mathbf{y}_i|h_i = r, \mathbf{t}; \boldsymbol{\Psi}_r), \tag{2.140}$$

where $\alpha_r = p(h_i = r)$ is the prior probability of the cluster r $(r = 1, \ldots, R)$, the class conditional distribution $p(\mathbf{y}_i|h_i = r, \mathbf{t}; \boldsymbol{\Psi}_r)$ is assumed to be an HMM, typically in this case of curves with unidimensional Gaussian emission probabilities, with parameters $\boldsymbol{\Psi}_r = (\boldsymbol{\pi}^r, \mathbf{A}^r, \mu_{r1}, , \ldots, \mu_{rK}, \sigma^2_{rk}, \ldots, \sigma^2_{rK})$ where $\boldsymbol{\pi}^r = (\pi^r_1, \ldots, \pi^r_K)$ is the initial distribution, $\mathbf{A}^r = (\mathbf{A}^r_{\ell k})_{1\leq \ell,k\leq K}$ is the transition matrix with and μ_{rk} and σ^2_{rk} for $r = 1, \ldots, R$ and $k = 1, \ldots, K$, are respectively the constant mean and the variance of an unidimensional Gaussian density. The distribution of a curve can therefore be defined in a similar way as in Equation (2.69). This model is therefore specified by the parameter vector

$$\boldsymbol{\Psi} = (\alpha_1, \ldots, \alpha_R, \boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_R).$$

Two different clustering formulations can be adopted for estimating this mixture of HMMs. Two such techniques are the hard-clustering k-means approach and the soft-clustering EM approach. A K-means-like approach for hard clustering is used in (Alon et al., 2003; Smyth, 1996) in which the optimized fucntion is the complete-data log-likelihood. The resulting clustering scheme consists of assigning sequences to clusters in each iteration and use only the sequences assigned to a cluster for re-estimation of its HMM parameters. The soft clustering approach is described in Alon et al. (2003) where the model parameters are estimated in a maximum likelihood framework by EM.

Notice that for the two approaches, one needs to use the Baum-Welch algorithm to compute the joint posterior probabilities for the HMM states and the conditional distribution (the HMM likelihood) for each curve through the forward-backward procedures. The parameter updating formulas are given in a closed form and can be found in Alon et al. (2003); Jebara et al. (2007).

2.8 Curve classification

In the previous section, we considered the problem of curve clustering in which a partition of the curves is automatically estimated from a set of curves, typically heterogeneous. In this part we describe some methods for curve classification. Curve classification arises when the class labels are available for a given training set, and we aim at predicting the class labels for new observed curves.

Curve classification problem have been already addressed using discriminative approaches which include neural network approaches (Rossi and Conan-Guez, 2005a) and kernel-based learning methods (Rossi and Villa, 2006) as well as generative approaches like spline regression (James and Hastie, 2001). Here we focus on generative approaches for modeling the classes of curves. They include polynomial regression, polynomial splines and B-splines, and polynomial piecewise regression. After the learning step, a class prediction can then be performed by using Bayes' theorem which yields to Functional Linear Discriminant Analysis (FLDA) as in (James and Hastie, 2001) where a natural cubic spline is used to model a set curves. This approach is analogous to Linear Discriminant Analysis in the case of multidimensional data. The principle of FLDA for different curve models is given in the next section.

2.8.1 Functional Linear Discriminant Analysis

Let us denote by $((\mathbf{y}_1, c_1), \ldots, (\mathbf{y}_n, c_n))$ a given labeled training set of curves issued from G classes with $(c_i \in \{1, \ldots, G\})$ is the class label of the ith curve. FLDA arises when we model each class of curves (functions) with a single model (e.g., polynomial, spline, B-spline or a piecewise function). Assume we have learned the class parameter vectors $(\hat{\boldsymbol{\Psi}}_1, \ldots, \hat{\boldsymbol{\Psi}}_G)$ from the labeled training set of curves, where $\hat{\boldsymbol{\Psi}}_g$ is an estimation of the parameter vector $\boldsymbol{\Psi}_g$ of class g $(g = 1, \ldots, G)$ provided by the corresponding estimation procedure (c.f., see section 2.6 for detail on the optimization procedure for these different models). A new curve $\mathbf{y}_i$ is then assigned to the class $\hat{c}_i$ using the MAP rule. Formally, the class of a new curve is given by

$$\hat{c}_i = \arg \max_{1 \leq g \leq G} p(c_i = g | \mathbf{y}_i, \mathbf{t}; \hat{\boldsymbol{\Psi}}_g), \tag{2.141}$$

where

$$p(c_i = g | \mathbf{y}_i, \mathbf{t}; \hat{\boldsymbol{\Psi}}_g) = \frac{p(c_i = g) p(\mathbf{y}_i | c_i = g, \mathbf{t}; \hat{\boldsymbol{\Psi}}_g)}{\sum_{g'=1}^{G} p(c_i = g') p(\mathbf{y}_i | c_i = g', \mathbf{t}; \hat{\boldsymbol{\Psi}}_{g'})}, \tag{2.142}$$

is the class posterior probability, $p(c_i = g)$ being the proportion of the class g in the training set and $p(\mathbf{y}_i | c_i = g, \mathbf{t}; \hat{\boldsymbol{\Psi}}_g)$ the conditional density of the class g defined by Equations (2.94) and (2.105).

It is worth to note that this mode is more suitable for homogeneous classes of curves. However, when one or more classes are dispersed, using a single model description becomes unsuitable. There exists an extension of FLDA called Functional Mixture Discriminant Analysis (FMDA) to handle the problem of sub-classes of curves.

2.8.2 Functional Mixture Discriminant Analysis

Motivated by the complexity of the gene functions, for which modeling each class with a single function using FLDA is not adapted, Gui and Li (2003) proposed a Functional Mixture Discriminant Analysis approach (FMDA) for funtional data classification by analogy to Mixture Discriminant Analysis (MDA) (Hastie and Tibshirani, 1996) for the case of multidimensional data. In the FMDA approach developed in Gui and Li (2003), each class of functions is modeled by a mixture of several sub-classes, each of the sub-classes is a B-spline function with Gaussian noise. The EM algorithm is used for estimating the parameters in a maximum likelihood framework.

Notice that the choice of the component density model is generic and can be assumed to be a spline function as well as a polynomial or a piecewise polynomial function. The density of a curve given the class (the group) g $(g = 1, \ldots, G)$ for FMDA is a mixture density which can be written as

$$p(\mathbf{y}_i|c_i = g, \mathbf{t}; \boldsymbol{\Psi}_g) \quad = \quad \sum_{r=1}^{R_g} \alpha_{gr}\; p(\mathbf{y}_i|c_i = g, h_i = r, \mathbf{t}; \boldsymbol{\Psi}_{gr})$$

where Rg is the number of component densities for class g, the α_{gr}'s $(r = 1, \ldots, R_g)$ are the corresponding non-negative mixing proportions that sum to 1 and the component density $p(\mathbf{y}_i|c_i = g, h_i = r, \mathbf{t}; \boldsymbol{\Psi}_{gr})$ parametrized by $\boldsymbol{\Psi}_{gr}$ can be defined by a polynomial regression or a polynomial spline (including B-spline) regression model (c.f., Equation (2.94)) or a polynomial piecewise regression model (c.f., Equation (2.105)). The parameters of the mixture model for each class g denoted by

$$\boldsymbol{\Psi}_g = (\alpha_{g1}, \ldots, \alpha_{gR_g}, \boldsymbol{\Psi}_{g1}, \ldots, \boldsymbol{\Psi}_{gR_g})$$

can be estimated by maximum likelihood using the EM algorithm Dempster et al. (1977). Indeed, estimating the parameters for each class is equivalent to performing a curve clustering for the set of curves belonging to this class, which was detailed in section 2.7. The number of component densities can be chosen by using some model selection criteria such as BIC or AIC. Note that in Gui and Li (2003) the number of sub-classes are fixed by the user.

2.9 Summary

In this chapter, we reviewed probabilistic models used for modeling, classification and clustering multidimensional data and functional data.We also considered both the static and the dynamic aspects. For the static aspect, the sample is assumed to be independent, and for the dynamic aspect, time dependency of the data can be inferred through dedicated models for sequence modeling.

For multidimensional data classification, MDA has shown good performance by providing accurate class separation as compared to LDA, QDA and logistic regression. For the case of sequential observations, although HMMs and ARHMMs are suitable to capture dynamical aspects of a system given an observation sequence, they assume a homogeneous Markov chain. Therefore, extending them to the non-homogeneous case should be beneficial for modeling time-varying state transitions.

The methods presented for curve modeling, clustering and classification are mainly model-based and rely on regression, piecewise regression and HMMs. However, the majority of these approaches do not address the problem of regime changes within curves. Indeed, the approaches based on polynomial regression or spline regression are not dedicated to regime change detection problems. Fitting several models to a curve or a set of curves for different time ranges is possible with a piecewise polynomial regression. However, this approach may be computationally expensive and does not guarantee the continuity of the estimated curve. Finally, although HMM regression can be used for modeling a single curve with regime changes, its formulation for a set of curves is not adapted for providing a single “mean curve” representation.

Chapter 3

Curve modeling by a regression model with a hidden logistic process

Contents

3.1 Introduction

This chapter focuses on modeling and classifying curves with regime changes. Figure 3.1 shows examples of curves issued from the railway switch operations. We focus on

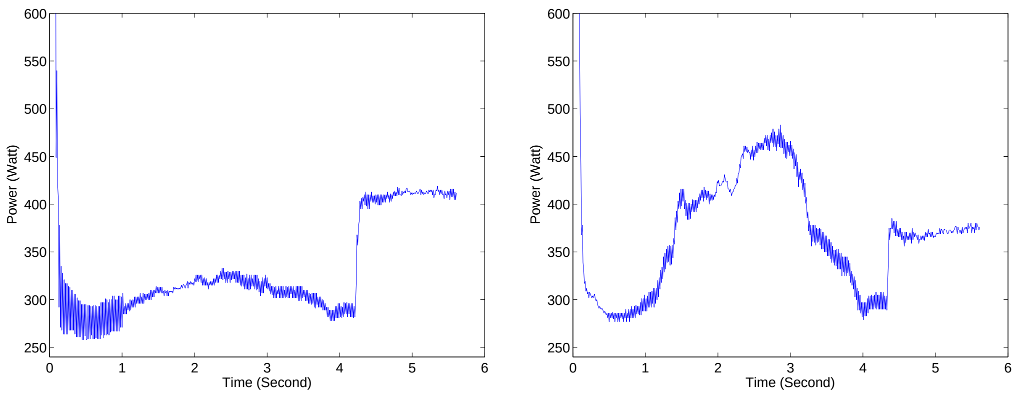


Figure 3.1: Example of curves of electrical power consumed during two switch operations.

curves that present non-linearities and various regime changes, which is the case for the switch operation curves that we are studying in this thesis (see Figure 3.1). The aim is to provide an accurate representation of such curves through a dedicated modeling approach. In particular, a parametric model that provides a feature vector of reduced dimension and, at the same time, preserves most of the relevant information concerned with the dynamical behavior of the regimes (including their temporal location, their shapes and their dispersion) should, without doubt, be very promising for curve classification and/or curve clustering tasks. More specifically, generative models are parametric models that help us understand the underlying processes generating such curves. We therefore opt for a parametric generative approach for curve modeling.

As shown in Figure 3.1, we have nonlinear regimes in which the changes over time may be smooth and/or abrupt. In such a context, basic parametric methods based on linear or polynomial regression are not appropriate. The piecewise regression model shown in 2 may be used as an alternative. Recall that piecewise polynomial regression is a modeling and segmentation method that partitions the curve into K segments or regimes, with each segment characterized by its mean polynomial curve and its variance. For this type of modeling, because the optimized criterion is additive over the segments (or regimes), parameter estimation can be performed using dynamic programming (Bellman, 1961; Stone, 1961). Another alternative approach is to use the Hidden Markov Model Regression (Fridman, 1993) in which model parameters are estimated using the Baum-Welch algorithm (Baum et al., 1970).

In this chapter, we present a new approach for curve modeling (Chamroukhi et al., 2009c). It consists of a regression model that incorporates a discrete hidden logistic process. We call this model RHLP. This approach is related to the switching regression model introduced by Quandt and Ramsey (1978) and is linked to the Mixture of Experts (ME) model developed by Jordan and Jacobs (1994) by the use of a time-dependent logistic transition function. The ME model, as discussed in Waterhouse (1997), uses a conditional mixture modeling in which model parameters are estimated using the EM algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997).

The proposed model allows for activating smoothly and/or abruptly different polynomial regression models over time. The model parameters are estimated by the maximum likelihood method performed with a dedicated Expectation-Maximization (EM) algorithm. Then, we apply the RHLP model for curve classification using Mixture Discriminant Analysis (MDA) (Hastie and Tibshirani, 1996). The contribution we present here involves curve modeling.

This chapter is organized as follows. Section 3.2 introduces the proposed regression model and the hidden logistic process. Section 3.3 details parameter estimation via a dedicated EM algorithm. In section 3.4, the RHLP method is applied to curve classification using Mixture Discriminant Analysis. Finally, in section 3.5 we provide illustrations on simulated curves using the proposed RHLP model.

3.2 The regression model with a hidden logistic process (RHLP)

The proposed regression model is defined, as for the HMMR model, by

$$y_j = \boldsymbol{\beta}_{z_j}^T \mathbf{t}_j + \sigma_{z_j}\epsilon_j \quad ; \quad \epsilon_j \sim \mathcal{N}(0,1), \quad (j = 1, \ldots, m) \tag{3.1}$$

where in this case a logistic process is used to model the hidden sequence $\mathbf{z} = (z_1, \ldots, z_m)$. In the next section we give the details of the proposed hidden logistic process and highlight its flexibility for modeling the dynamical behavior within a curve through accurately capturing the regime changes.

3.2.1 The hidden logistic process

This section defines the probability distribution of the process $\mathbf{z} = (z_1, \ldots, z_m)$ that allows for the switching from one regression model to another.

The proposed hidden logistic process assumes that the variables z_j $(j = 1, \ldots, m)$, given the vector $\boldsymbol{t} = (t_1, \ldots, t_m)$, are generated independently according to the multinomial distribution $\mathcal{M}(1, \pi_1(t_j; \mathbf{w}), \ldots, \pi_K(t_j; \mathbf{w}))$, where

$$\pi_k(t_j; \mathbf{w}) = p(z_j = k|t_j; \mathbf{w}) = \frac{\exp\left(\boldsymbol{w}_k^T \boldsymbol{v}_j\right)}{\sum_{\ell=1}^K \exp\left(\boldsymbol{w}_\ell^T \boldsymbol{v}_j\right)}, \tag{3.2}$$

is the logistic transformation of a linear function of the time-dependent covariate vector $\boldsymbol{v}_j = (1, t_j, t_j^2, \ldots, t_j^u)^T$, $\boldsymbol{w}_k = (\boldsymbol{w}_{k0}, \ldots, \boldsymbol{w}_{ku})^T$ is the $(u+1)$-dimensional coefficients vector associated with $\boldsymbol{v}_j$ and $\mathbf{w} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K)$. Thus, given the vector $\boldsymbol{t} = (t_1, \ldots, t_m)$, the distribution of $\mathbf{z}$ can be written as

$$p(\mathbf{z}|\mathbf{t}; \mathbf{w}) = \prod_{j=1}^m \prod_{k=1}^K p(z_j = k|t_j; \mathbf{w})^{z_{jk}} = \prod_{j=1}^m \prod_{k=1}^K \left(\frac{\exp\left(\boldsymbol{w}_k^T \boldsymbol{v}_j\right)}{\sum_{\ell=1}^K \exp\left(\boldsymbol{w}_\ell^T \boldsymbol{v}_j\right)}\right)^{z_{jk}}, \tag{3.3}$$

where z_{jk} is an indicator-binary variable such that $z_{jk} = 1$ if $z_j = k$ (i.e., when y_j is generated by the kth regression model), and 0 otherwise.

The relevance of the logistic transformation in terms of flexibility of transitions can be illustrated through simple examples with $K = 2$ components. In this case, only the probability $\pi_1(t_j; \mathbf{w}) = \frac{exp(\boldsymbol{w}_1^T \boldsymbol{v}_j)}{1+exp(\boldsymbol{w}_1^T \boldsymbol{v}_j)}$ should be described, since $\pi_2(t_j; \mathbf{w}) = 1 - \pi_1(t_j; \mathbf{w})$. The first example is designed to show the effect of the dimension u of $\boldsymbol{w}_k$ on the temporal variation of the probabilities $\pi_k(t_j; \mathbf{w})$. We consider different values of the dimension $u$of $\boldsymbol{w}_k$ $(u = 0, 1, 2)$.

As shown in Figure 3.2, the dimension u controls the number of temporal transitions of $\pi_k(t_j; \mathbf{w})$. In fact, the larger the dimension of $\boldsymbol{w}_k$, the more complex the temporal variation of $\pi_k(t_j; \mathbf{w})$. More particularly, if the goal is to segment the curves into contiguous segments, the dimension u of $\boldsymbol{w}_k$ must be set to 1, what will be assumed hereafter.

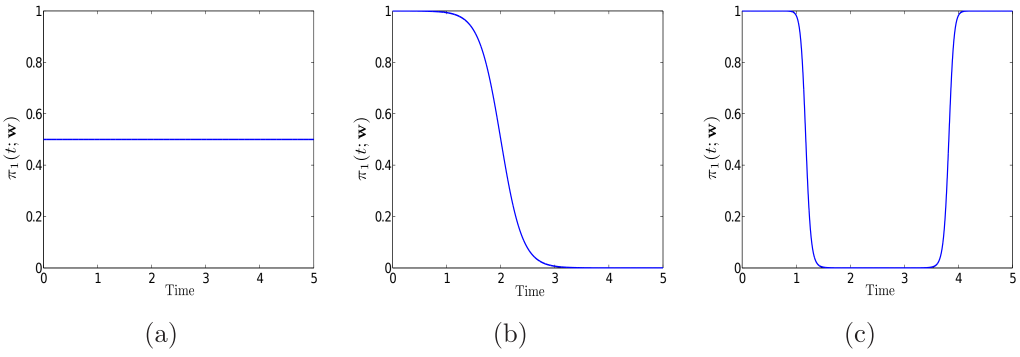


Figure 3.2: Variation of $\pi_1(t_j; \mathbf{w})$ over time for different values of the dimension q of $\boldsymbol{w}_1$, for $K = 2$ and (a) $u = 0$ and $\boldsymbol{w}_1 = 0$, (b) $u = 1$ and $\boldsymbol{w}_1 = (10, -5)^T$ and (c) $u = 2$ and $\boldsymbol{w}_1 = (-10, -20, -4)^T$.

For a fixed dimension u of the parameter $\boldsymbol{w}_k$, the variation of the proportions $\pi_k(t_j; \mathbf{w})$ over time, in relation to the parameter $\boldsymbol{w}_k$, is illustrated by an example of 2 regimes with $u = 1$. For this purpose, we use the parametrization $\boldsymbol{w}_k = \lambda_k(\gamma_k, 1)^T$ of $\boldsymbol{w}_k$, where $\lambda_k = \boldsymbol{w}_{k1}$ and $\gamma_k = \frac{\boldsymbol{w}_{k0}}{\boldsymbol{w}_{k1}}$. As shown in Figure 3.3 (a), the parameter λ_k controls the quality of transitions between the regimes, the higher absolute value of λ_k, the more abrupt the transition between the regimes z_j. Whereas the parameter γ_k, as it can be seen in Figure 3.3 (b), is directly related to the placement of the transition time point.

In this particular regression model, the variable z_j controls the switching from one regression model to another of K regression models at each time t_j. Therefore, unlike basic polynomial regression models which can be seen as stationary models as they assume uniform regression parameters over time, the proposed dynamical regression model allows for polynomial coefficients to vary over time by switching from one regression model to another. This modeling is therefore beneficial for capturing non-stationary behavior involved by regime changes for a curve.

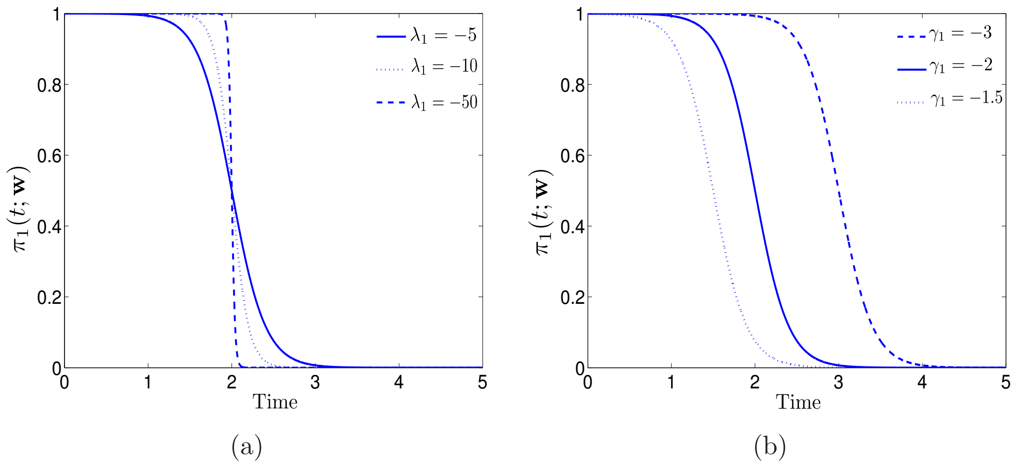


Figure 3.3: Variation of $\pi_1(t_j;\mathbf{w})$ over time for a dimension $u=1$ of $\boldsymbol{w}_1$ and (a) different values of $\lambda_1=\boldsymbol{w}_{11}$ with $\gamma_1=-2$ and (b) different values of $\gamma_1=\frac{\boldsymbol{w}_{10}}{\boldsymbol{w}_{11}}$ with $\lambda_1=-5$.

3.2.2 The generative model

The generative model for a curve $\mathbf{y}_i=(y_1,\ldots,y_m)$ given the parameter vector $\boldsymbol{\theta}=(\mathbf{w},\boldsymbol{\beta}_1,\ldots,\boldsymbol{\beta}_K,\sigma_1^2,\ldots,\sigma_K^2)$ consists of 2 steps:

- generate the hidden process $\mathbf{z}=(z_1,\ldots,z_m)$ according to the multinomial distribution $z_j|t_j\sim\mathcal{M}(1,\pi_1(t_j;\mathbf{w}),\ldots,\pi_K(t_j;\mathbf{w}))$,
- generate each observation y_j according to the Gaussian distribution $\mathcal{N}(\cdot;\boldsymbol{\beta}_{z_j}^T\mathbf{t}_j,\sigma_{z_j}^2)$.

A graphical representation for the RHLP model is presented in Figure 3.4.

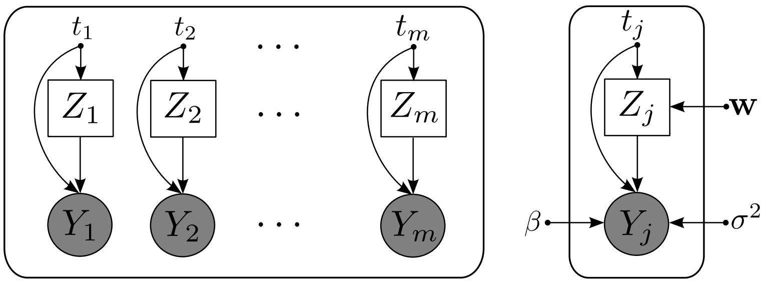


Figure 3.4: Graphical model structure for the proposed regression model with a hidden logistic process (RHLP).

3.3 Parameter estimation by a dedicated EM algorithm

For the proposed model, conditionally on a regression model k, y_j is distributed according to a normal density with mean $\boldsymbol{\beta}_k^T\mathbf{t}_j$ and variance σ_k^2:

$$p(y_j|z_j=k,t_j;\boldsymbol{\theta}_k)=\mathcal{N}\big(y_j;\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\big) \qquad (3.4)$$

where $\boldsymbol{\theta}_k = (\boldsymbol{\beta}_k, \sigma_k^2)$. Thus, the observation y_j at each time point t_j is distributed according to the following normal mixture density:

$$\begin{aligned} p(y_j|t_j;\boldsymbol{\theta}) &= \sum_{k=1}^{K} p(z_j = k|t_j;\mathbf{w})p(y_j|z_j = k, t_j;\boldsymbol{\theta}_k) \\ &= \sum_{k=1}^{K} \pi_k(t_j;\mathbf{w})\mathcal{N}\big(y_j;\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\big), \end{aligned} \tag{3.5}$$

where $\boldsymbol{\theta} = (\mathbf{w}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \sigma_1^2, \ldots, \sigma_K^2)$ is the unknown parameter vector to be estimated. The parameter $\boldsymbol{\theta}$ is estimated by the maximum likelihood method. As in the classic regression models we assume that, given $\boldsymbol{t} = (t_1, \ldots, t_m)$, the ϵ_j are independent. This also implies the independence of y_j $(j = 1, \ldots, m)$ given the time vector $\mathbf{t}$. The log-likelihood of $\boldsymbol{\theta}$ is therefore written as

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta};\mathbf{y},\mathbf{t}) &= \log \prod_{j=1}^{m} p(y_j|t_j;\boldsymbol{\theta}) \\ &= \sum_{j=1}^{m} \log \sum_{k=1}^{K} \pi_k(t_j;\mathbf{w})\mathcal{N}\big(y_j;\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\big). \end{aligned} \tag{3.6}$$

The maximization of this log-likelihood can not be performed in a closed form as it results in a complex nonlinear function due to the logarithm of the sum. However, in this latent data framework, the Expectation-Maximization (EM) algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997) is particularly adapted for maximizing the log-likelihood. With this specification, given a particular configuration $\mathbf{z}$ of the hidden process, the complete-data log-likelihood of $\boldsymbol{\theta}$ is given by:

$$\mathcal{L}_c(\boldsymbol{\theta};\mathbf{y},\mathbf{z}) = \log p(\mathbf{y},\mathbf{z}|\mathbf{t};\boldsymbol{\theta}) = \log[p(\mathbf{y}|\mathbf{z},\mathbf{t};\boldsymbol{\theta})p(\mathbf{z}|\mathbf{t};\mathbf{w})]. \tag{3.7}$$

Since for a particular configuration of the hidden process $\mathbf{z}$, the conditional distribution of the observed curve $\mathbf{y}$ given the vector of time points $\mathbf{t}$ is given by:

$$\begin{aligned} p(\mathbf{y}|\mathbf{z},\mathbf{t};\boldsymbol{\theta}) &= \prod_{j=1}^{m} p(y_j|z_j, t_j;\boldsymbol{\theta}) \\ &= \prod_{j=1}^{m}\prod_{k=1}^{K} p(y_j|z_j = k, t_j;\boldsymbol{\theta}_k)^{z_{jk}}, \end{aligned} \tag{3.8}$$

thus, after collecting together (4.4) and (3.8) in (3.7), we then obtain the expression of the complete-data log-likelihood (3.7) :

$$\mathcal{L}_c(\boldsymbol{\theta};\mathbf{y},\mathbf{z},\mathbf{t}) = \sum_{j=1}^{m}\sum_{k=1}^{K} z_{jk} \log\left[\pi_k(t_j;\mathbf{w})\mathcal{N}\left(y_j;\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\right)\right]. \tag{3.9}$$

The next section presents the proposed EM algorithm for the regression model with hidden logistic process (RHLP).

3.3.1 The dedicated EM algorithm

The proposed EM algorithm starts with an initial parameter $\boldsymbol{\theta}^{(0)}$ and alternates between the two following steps until convergence:

E-Step: This step consists of computing the expectation of the complete-data log-likelihood (3.9) , given the observations and the current value $\boldsymbol{\theta}^{(q)}$ of the parameter $\boldsymbol{\theta}$ (q being the current iteration):

$$\begin{aligned}
Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(q)}) &= \mathbb{E}\left[\mathcal{L}_c(\boldsymbol{\theta}; \mathbf{y}, \mathbf{t}, \mathbf{z})|\mathbf{y}, \mathbf{t}; \boldsymbol{\theta}^{(q)}\right] \\
&= \sum_{j=1}^{m}\sum_{k=1}^{K} \mathbb{E}[z_{jk}|y_j, t_j; \boldsymbol{\theta}^{(q)}] \log\left[\pi_k(t_j;\mathbf{w})\mathcal{N}(y_j; \boldsymbol{\beta}_k^T\mathbf{t}_j, \sigma_k^2)\right] \\
&= \sum_{j=1}^{m}\sum_{k=1}^{K} \tau_{jk}^{(q)} \log\left[\pi_k(t_j;\mathbf{w})\mathcal{N}\left(y_j; \boldsymbol{\beta}_k^T\mathbf{t}_j, \sigma_k^2\right)\right] \\
&= \sum_{j=1}^{m}\sum_{k=1}^{K} \tau_{jk}^{(q)} \log \pi_k(t_j;\mathbf{w}) + \sum_{j=1}^{m}\sum_{k=1}^{K} \tau_{jk}^{(q)} \log \mathcal{N}\left(y_j; \boldsymbol{\beta}_k^T\mathbf{t}_j, \sigma_k^2\right), \quad (3.10)
\end{aligned}$$

where

$$\begin{aligned}
\tau_{jk}^{(q)} &= p(z_{jk}=1|y_j, t_j; \boldsymbol{\theta}^{(q)}) \\
&= \frac{\pi_k(t_j;\mathbf{w}^{(q)})\mathcal{N}(y_j; \boldsymbol{\beta}_k^{T(q)}\mathbf{t}_j, \sigma_k^{2(q)})}{\sum_{\ell=1}^{K}\pi_\ell(t_j;\mathbf{w}^{(q)})\mathcal{N}(y_j; \boldsymbol{\beta}_\ell^{T(q)}\mathbf{t}_j, \sigma_\ell^{2(q)})} \quad (3.11)
\end{aligned}$$

is the posterior probability that y_j originates from the kth polynomial regression model. As shown in the expression of the Q-function, this step simply requires the computation of the posterior probabilities $\tau_{jk}^{(q)}$.

M-Step: In this step, the value of the parameter $\boldsymbol{\theta}$ is updated by computing the parameter $\boldsymbol{\theta}^{(q+1)}$ maximizing the conditional expectation Q with respect to $\boldsymbol{\theta}$:

$$\boldsymbol{\theta}^{(q+1)} = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(q)}) \quad (3.12)$$

where $\boldsymbol{\Theta}$ is the parameter space.

Let us denote by $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$ the term in $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(q)})$ that is function of $\mathbf{w}$ and by $Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(q)})$ the term in $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(q)})$ that depends on $\boldsymbol{\theta}_k = (\boldsymbol{\beta}_k, \sigma_k^2)$, we obtain:

$$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(q)}) = Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)}) + \sum_{k=1}^{K} Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta}, \boldsymbol{\theta}^{(q)}), \quad (3.13)$$

where

$$Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)}) = \sum_{j=1}^{m}\sum_{k=1}^{K} \tau_{jk}^{(q)} \log \pi_k(t_j;\mathbf{w}), \quad (3.14)$$

and

$$\begin{aligned} Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)}) &= \sum_{j=1}^{m}\tau_{jk}^{(q)}\log\mathcal{N}\left(y_j;\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\right)\\ &= -\frac{1}{2}\Big[\frac{1}{\sigma_k^2}\sum_{j=1}^{m}\tau_{jk}^{(q)}\left(y_j-\boldsymbol{\beta}_k^T\mathbf{t}_j\right)^2+m_k^{(q)}\log\sigma_k^2\Big]-\frac{m_k^{(q)}}{2}\log 2\pi \quad (3.15)\end{aligned}$$

for $k=1,\ldots,K$, where $m_k^{(q)}=\sum_{j=1}^{m}\tau_{jk}^{(q)}$ can be seen as the number of points of the component k estimated at the iteration q. Thus, the maximization of Q with respect to $\boldsymbol{\theta}$ can be performed by separately maximizing $Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})$ with respect to $\mathbf{w}$ and $Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})$ with respect to $(\boldsymbol{\beta}_k,\sigma_k^2)$ for all $k=1,\ldots,K$.

In order to maximize $Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})$ with respect to the regression coefficients $\boldsymbol{\beta}_k$, let us denote by $Q_{\boldsymbol{\beta}_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})$ the term in $Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})$ that references the regression coefficients $\boldsymbol{\beta}_k$, we get:

$$Q_{\boldsymbol{\beta}_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})=-\frac{1}{2\sigma_k^2}\sum_{j=1}^{m}\tau_{jk}^{(q)}\left(y_j-\boldsymbol{\beta}_k^T\mathbf{t}_j\right)^2. \quad (3.16)$$

Thus, maximizing $Q_{\boldsymbol{\beta}_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})$ w.r.t $\boldsymbol{\beta}_k$ consists of solving the weighted least-squares problem where the weights are the posterior probabilities $\tau_{jjk}^{(q)}$

$$\boldsymbol{\beta}_k^{(q+1)}=\arg\min_{\boldsymbol{\beta}_k\in\mathbb{R}^{p+1}}\sum_{j=1}^{m}\tau_{jk}^{(q)}(y_j-\boldsymbol{\beta}_k^T\mathbf{t}_j)^2.$$

The solution of this problem is obtained in a closed form from the so-called normal equations and is given by:

$$\begin{aligned}\boldsymbol{\beta}_k^{(q+1)} &= \Big[\sum_{j=1}^{m}\tau_{jk}^{(q)}\mathbf{t}_j\mathbf{t}_j^T\Big]^{-1}\sum_{j=1}^{m}\tau_{jk}^{(q)}y_j\mathbf{t}_j\\ &= (\mathbf{X}^T\mathbf{W}_k^{(q)}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}_k^{(q)}\mathbf{y}, \quad (3.17)\end{aligned}$$

where $\mathbf{W}_k^{(q)}$ is an $m\times m$ diagonal matrix of weights whose diagonal elements are $(\tau_{1k}^{(q)},\ldots,\tau_{mk}^{(q)})$ and $\mathbf{X}$ is the $n\times(p+1)$ regression matrix.

To update the estimation of the variance σ_k^2 for each of the K polynomial regimes, letting $Q_{\sigma_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})$ denotes the terms in $Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})$ that are function of σ_k^2, we obtain:

$$Q_{\sigma_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)}) = -\frac{1}{2}\sum_{j=1}^{m}\tau_{jk}^{(q)}\Big[\frac{(y_j-\boldsymbol{\beta}_k^{T(q+1)}\mathbf{t}_j)^2}{\sigma_k^2}+\log\sigma_k^2\Big]. \quad (3.18)$$

Thus, maximizing $Q_{\sigma_k}(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})$ w.r.t σ_k^2 is a weighted variant of the problem of estimating the variance of an univariate Gaussian density. The problem can be solved in a closed form. Taking the derivative of (3.18) with respect to σ_k^2 and setting to zero yields:

$$\begin{aligned}\sigma_k^{2(q+1)} &= \frac{1}{m_k^{(q)}}\sum_{j=1}^{m}\tau_{jk}^{(q)}(y_j - \boldsymbol{\beta}_k^{T(q+1)}\mathbf{t}_j)^2\\ &= \frac{1}{m_k^{(q)}}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}_k^{(q+1)})^T\mathbf{W}_k^{(q)}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta}_k^{(q+1)}). \qquad (3.19)\end{aligned}$$

The maximization of $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$ with respect to $\mathbf{w}$ is a multinomial logistic regression problem weighted by $\tau_{jk}^{(q)}$ which we solve with a multi-class Iterative Reweighted Least Squares (IRLS) algorithm (Chamroukhi et al., 2009a; Chen et al., 1999; Green, 1984; Krishnapuram et al., 2005). The IRLS algorithm is detailed in the following paragraph.

The Iteratively Reweighted Least Squares (IRLS) algorithm: The IRLS algorithm is used to maximize

$$Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)}) = \sum_{j=1}^{m}\sum_{k=1}^{K}\tau_{jk}^{(q)}\log\pi_k(t_j;\mathbf{w})$$

with respect to the parameter $\mathbf{w}$ in the M step at each iteration q of the EM algorithm. This criterion is concave. We derive the concavity proof in Appendix A.7. To estimate the parameter vector $\mathbf{w} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K)$, the component vector $\boldsymbol{w}_K$ is set to the null vector to satisfy $\sum_{k=1}^{K}\pi_k(t_j;\mathbf{w}) = 1$. The IRLS algorithm is equivalent to the Newton-Raphson algorithm, which consists of starting with a vector $\mathbf{w}^{(0)}$, and updating the estimation of $\mathbf{w}$ as follows:

$$\mathbf{w}^{(l+1)} = \mathbf{w}^{(l)} - \left[\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})}{\partial\mathbf{w}\partial\mathbf{w}^T}\right]^{-1}_{\mathbf{w}=\mathbf{w}^{(l)}}\frac{\partial Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})}{\partial\mathbf{w}}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} \qquad (3.20)$$

where $\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})}{\partial\mathbf{w}\partial\mathbf{w}^T}$ and $\frac{\partial Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})}{\partial\mathbf{w}}$ are respectively the Hessian matrix and the gradient of $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$. At each IRLS iteration the Hessian and the gradient are evaluated at $\mathbf{w} = \mathbf{w}^{(l)}$ and are computed similarly as in the IRLS algorithm described in section 2.4.4, by replacing the binary membership variables y_{ig} and the logistic probabilities $\pi_g(\mathbf{x}_i;\mathbf{w})$ in both Equation (2.55) and Equation (2.58), by the posterior probabilities τ_{jk} and the logistic probabilities $\pi_k(t_j;\mathbf{w})$. The calculation details are also given in Appendix A.7.

The parameter update $\mathbf{w}^{(q+1)}$ is taken at convergence of the IRLS algorithm (3.20).

One can limit the number of iterations of the IRLS procedure of the EM algorithm. This version would consist of only increasing the criterion $Q_{\mathbf{w}}(\mathbf{w}, \mathbf{w}^{(q)})$ at each EM iteration rather than maximizing it. One can, for example, limit the number of IRLS iterations up to a single iteration. This scheme yields to a Generalized EM (GEM) algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997) which has the same convergence properties as the EM algorithm. However, in practice, we observed that this restriction results in an increase of the total EM iterations. We opted for the following initialization strategy. The IRLS procedure is initialized randomly only for the first EM iteration. For this first initialization, the convergence of the IRLS algorithm requires about fifteen iterations (recall that the main advantage of the IRLS algorithm, as a Newton Raphson procedure has a quadratic convergence rate (Boyd and Vandenberghe, 2004)). From the second iteration of the EM algorithm, the IRLS algorithm defined by Equation (3.20) is initialized with the parameter $\mathbf{w}^{(q)}$ estimated at the previous EM iteration. Beyond the fourth iteration of the EM algorithm, we observed that the IRLS algorithm converges after only about 5 iterations.

The time complexity of the E-step of this EM algorithm is of $\mathcal{O}(Km)$. The calculation of the regression coefficients in the M-step requires the computation and the inversion of the square matrix $\mathbf{X}^T\mathbf{X}$ which is of dimension $p+1$, and a multiplication by the observation sequence of length m which has a time complexity of $\mathcal{O}(p^2m)$. In addition, each IRLS loop requires an inversion of the Hessian matrix which is of dimension $(u+1)\times(K-1)$. Therefore for a small u (here we used $u=1$), the complexity of the IRLS loop is approximatively of $\mathcal{O}(I_{\text{IRLS}}K^2)$ where I_{IRLS} is the average number of iterations required by the internal IRLS algorithm. Therefore the proposed algorithm is performed with a time complexity of $\mathcal{O}(I_{\text{EM}}I_{\text{IRLS}}K^3p^2m)$, where I_{EM} is the number of iterations of the EM algorithm.

The pseudo code 3 summarizes one run of the proposed EM algorithm.

Algorithm 3 Pseudo code of the proposed algorithm for the RHLP model.

Inputs: a curve $\mathbf{y}$, the number of polynomial components K, the polynomial degree p and the sampling time $t_1, \ldots, t_m$.

```
 1: Initialize: $\boldsymbol{\theta}^{(0)} = (\mathbf{w}^{(0)}, \boldsymbol{\beta}_1^{(0)}, \ldots, \boldsymbol{\beta}_K^{(0)}, \sigma_1^{2(0)}, \ldots, \sigma_K^{2(0)})$
 2: fix a threshold $\epsilon > 0$
 3: set $q \leftarrow 0$ (EM iteration)
 4: while increment in log-likelihood $> \epsilon$ do
 5:    E-step:
 6:    for $k = 1, \ldots, K$ do
 7:       compute $\tau_{jk}^{(q)}$ for $j = 1, \ldots, m$ using equation (3.11)
 8:    end for
 9:    M-step:
10:    for $k = 1, \ldots, K$ do
11:       compute $\boldsymbol{\beta}_k^{(q+1)}$ using equation (3.17)
12:       compute $\sigma_k^{2(q+1)}$ using equation (3.19)
13:    end for
14:    IRLS:
15:    Initialize: set $\mathbf{w}^{(l)} = \mathbf{w}^{(q)}$
16:    set a threshold $\delta > 0$
17:    $l \leftarrow 0$ (IRLS iteration)
18:    while increment in $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)}) > \delta$ do
19:       compute $\mathbf{w}^{(l+1)}$ using equation (3.20)
20:       $l \leftarrow it + 1$
21:    end while
22:    $\mathbf{w}^{(q+1)} \leftarrow \mathbf{w}^{(l)}$
23:    $q \leftarrow q + 1$
24: end while
25: $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(q)}$
```

Output: $\hat{\boldsymbol{\theta}} = (\hat{\mathbf{w}}, \hat{\boldsymbol{\beta}}_1, \ldots, \hat{\boldsymbol{\beta}}_K, \hat{\sigma}_1^2, \ldots, \hat{\sigma}_K^2)$

3.3.2 Curve approximation and segmentation

In addition to performing curve modeling, the proposed approach can be used to approximate (or denoise) and segment curves. The curve approximation is given by the expectation vector $\mathbb{E}[\mathbf{y}|\mathbf{t};\hat{\boldsymbol{\theta}}] = \big(\mathbb{E}[y_1|t_1;\hat{\boldsymbol{\theta}}],\ldots,\mathbb{E}[y_m|t_m;\hat{\boldsymbol{\theta}}]\big)$, where

$$\begin{aligned}
\mathbb{E}[y_j|t_j;\hat{\boldsymbol{\theta}}] &= \int_{\mathbb{R}} y_j \; p(y_j|t_j;\hat{\boldsymbol{\theta}})dy_j \\
&= \int_{\mathbb{R}} y_j \sum_{k=1}^{K} \pi_k(t_j;\hat{\mathbf{w}})\mathcal{N}\big(y_j;\hat{\boldsymbol{\beta}}_k^T\mathbf{t}_j,\hat{\sigma}_k^2\big)dy_j \\
&= \sum_{k=1}^{K} \pi_k(t_j;\hat{\mathbf{w}}) \int_{\mathbb{R}} y_j \; \mathcal{N}\big(y_j;\hat{\boldsymbol{\beta}}_k^T\mathbf{t}_j,\hat{\sigma}_k^2\big)dy_j \\
&= \sum_{k=1}^{K} \pi_k(t_j;\hat{\mathbf{w}})\hat{\boldsymbol{\beta}}_k^T\mathbf{t}_j \;\; ,\forall j=1,\ldots,m, \qquad (3.21)
\end{aligned}$$

and $\hat{\boldsymbol{\theta}} = (\hat{\mathbf{w}},\hat{\boldsymbol{\beta}}_1,\ldots,\hat{\boldsymbol{\beta}}_K,\hat{\sigma}_1^2,\ldots,\hat{\sigma}_K^2)$ is the parameter vector obtained at convergence of the EM algorithm. The matrix formulation of the curve approximation equation can be written as

$$\hat{\mathbf{y}} = \sum_{k=1}^{K} \hat{\boldsymbol{\Pi}}_k\mathbf{X}\hat{\boldsymbol{\beta}}_k, \qquad (3.22)$$

where $\hat{\boldsymbol{\Pi}}_k$ is an $(m \times m)$ diagonal matrix whose diagonal elements are the logistic proportions $(\pi_k(t_1;\hat{\mathbf{w}}),\ldots,\pi_k(t_m;\hat{\mathbf{w}}))$ associated with the kth regression model.

On the other hand, a curve segmentation can also be obtained by computing the estimated label $\hat{z}_j$ of the polynomial regime generating y_j according to the following rule:

$$\hat{z}_j = \arg\max_{1\leq k\leq K} \pi_k(t_j;\hat{\mathbf{w}}), \quad (j=1,\ldots,m). \qquad (3.23)$$

Applying this rule guarantees the curves are segmented into contiguous segments if the probabilities π_k are computed with a dimension $u=1$ of $\boldsymbol{w}_k$ $(k=1,\ldots,K)$. This result lies in the fact that with this specification on the dimension the vector $\boldsymbol{w}_k$, and at the same time on the covariate vector $\boldsymbol{v}_j$, the separation between the polynomial regimes is linear in t. Indeed, the "decision boundary" between two regression models indexed by k and ℓ is represented by the set of time points for which we have $\pi_k(t_j;\mathbf{w}) = \pi_\ell(t_j;\mathbf{w})$ or by equivalence the log-ratios $\log\frac{\pi_k(t_j;\mathbf{w})}{\pi_\ell(t_j;\mathbf{w})}$ are zero, that is:

$$\log\frac{\pi_k(t_j;\mathbf{w})}{\pi_\ell(t_j;\mathbf{w})} = \log\frac{\exp(\boldsymbol{w}_k^T\boldsymbol{v}_j)}{\exp(\boldsymbol{w}_\ell^T\boldsymbol{v}_j)} = (\boldsymbol{w}_k - \boldsymbol{w}_\ell)^T\boldsymbol{v}_j = 0.$$

Let us recall that $\boldsymbol{v}_j = (1,t_j,\ldots,t_j^u)^T$. Therefore, if we set $u=1$ we have $\boldsymbol{v}_j = (1,t_j)^T$ and the obtained "decision boundary", which corresponds in this case to the set of transition time points t_j satisfying

$$(w_{k0} - w_{\ell 0}) + (w_{k1} - w_{\ell 1})t_j = 0$$

is linear in t_j. We therefore have a single transition point between the polynomial regimes k and ℓ given by

$$t_j = \frac{w_{k0} - w_{\ell 0}}{w_{\ell 1} - w_{k1}}.$$

When the degree u increases, the decision boundaries become nonlinear and arise in various transition time points (for example for $u = 2$ we can have 2 transition time points, etc (see Figure 3.2)), and therefore having contiguous segments will not be ensured.

3.3.3 Model selection

In a general application of the proposed model, the optimal values of (K, p, q) can be computed by using the Bayesian Information Criterion (BIC) (Schwarz, 1978) which is a penalized likelihood criterion, defined by

$$\mathrm{BIC}(K, p, u) = \mathcal{L}(\hat{\boldsymbol{\theta}}) - \frac{\nu_{\boldsymbol{\theta}} \log(m)}{2} \;, \tag{3.24}$$

where $\nu_{\boldsymbol{\theta}} = K(p + u + 3) - (u + 1)$ is the number of free parameters of the model and $\mathcal{L}(\hat{\boldsymbol{\theta}})$ is the observed-data log-likelihood obtained at convergence of the EM algorithm. To ensure a curve segmentation into contiguous polynomial segments u must be set to 1 which we adopt in the rest of this thesis. The dimension of the parameter space of in that case is given by $\nu_{\boldsymbol{\theta}} = K(p + 4) - 2$.

3.4 Application to curve classification

While the first objective of the RHLP model is to give a synthetic model that represents at best a rough curve, on the other hand, it is also a feature extraction technique for dimensionality reduction of the rough curves. Indeed, the RHLP model, given an observed curve, provides a parameter vector of dimension much lower than the number of total points necessary to describe the rough curve. In addition, this feature extraction technique does not lead to a loss in information that can has a strong effect on the resulting curve model as it can be seen through the examples shown in Figure 3.5 and Figure 3.6. In this way, the RHLP model is also promising for an accurate curve classification, based on the extracted features. The curve classifier to be built may be chosen from the existing machinery for multidimensional classification (supervised or unsupervised).

In the following sections we will therefore show how one can perform curve classification in both the supervised and the unsupervised contexts.

3.4.1 Modeling the curves with mixture models: a two steps strategy

The curve classification scheme is two-fold. Given a training set of n independent curves $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ where, the RHLP model detailed above is first applied to each of n curves. The estimated parameter vector for each curve is then used as its feature vector. The classification task is then performed in the resulting feature space (space of descriptors).

For the unsupervised curve classification context, we use the mixture model-based clustering. In this approach, given a training set of extracted feature vectors $(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)$, where $\boldsymbol{y}_i$ denotes the feature vector $\boldsymbol{\theta}$ extracted from the curve $\mathbf{y}_i$, the density for $\boldsymbol{y}_i$ is modeled by a Gaussian mixture distribution defined by:

$$p(\boldsymbol{y}_i; \boldsymbol{\Psi}) = \sum_{r=1}^{R} \alpha_r \mathcal{N}\left(\boldsymbol{y}_i; \boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r\right), \tag{3.25}$$

where

$$\boldsymbol{\Psi} = (\alpha_1, \ldots, \alpha_R, \boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_R)$$

is the parameter vector of the mixture, $\boldsymbol{\Psi}_r = (\boldsymbol{\mu}_r, \boldsymbol{\Sigma}_r)$ being the parameters of the Gaussian component r, and the corresponding non negative mixing proportions α_r ($r = 1, \ldots, R$) satisfy $\sum_{r=1}^{R} \alpha_r = 1$. The optimal number of Gaussian distributions R can be computed by maximizing the BIC criterion (Schwarz, 1978):

$$\mathrm{BIC}(R) = \mathcal{L}(\hat{\boldsymbol{\Psi}}) - \frac{\nu_R}{2} \log(n), \tag{3.26}$$

where $\hat{\boldsymbol{\Psi}}$ is the maximum likelihood estimate of $\boldsymbol{\Psi}$ provided by the EM algorithm and $\nu_R = \frac{R(\nu_{\boldsymbol{\theta}}+1)(\nu_{\boldsymbol{\theta}}+2)}{2} - 1$ is the dimension of $\boldsymbol{\Psi}$.

Estimating a partition for the curves using the MAP rule

Given the estimated model parameters $\hat{\boldsymbol{\Psi}}$ from a training set of n curves by, a partition of the curves into R clusters can then be estimated by using the MAP rule. According to this rule, the cluster label of the ith curve is calculated as:

$$\hat{h}_i = \arg \max_{1 \leq r \leq R} p(h_i = r | \boldsymbol{y}_i; \hat{\boldsymbol{\Psi}}_r), \tag{3.27}$$

where

$$p(h_i = r | \boldsymbol{y}_i; \hat{\boldsymbol{\Psi}}_r) = \frac{\hat{\alpha}_r \mathcal{N}(\boldsymbol{y}_i; \hat{\boldsymbol{\Psi}}_r)}{\sum_{r'=1}^{R} \hat{\alpha}_{r'}\, p(\boldsymbol{y}_i; \hat{\boldsymbol{\Psi}}_{r'})} \tag{3.28}$$

is the posterior probability of cluster r.

3.4.2 Application to curve classification with MDA

This part is devoted to the application of the RHLP model to curve classification into predefined classes using Mixture Discriminant Analysis.

To perform the curve classification, we opt for Mixture Discriminant Analysis (Hastie and Tibshirani, 1996) which allows for complex decision boundaries and is able to handle the class dispersion problem. Given a training set of a labeled collection of extracted feature vectors $((\boldsymbol{y}_1, c_1), \ldots, (\boldsymbol{y}_n, c_n))$, where $\boldsymbol{y}_i$ denotes the feature vector $\boldsymbol{\theta}$ extracted from the curve $\mathbf{y}_i$, the parameters of each class are learned using Mixture Discriminant Analysis (MDA) (Hastie and Tibshirani, 1996). In this approach, the density of each class g ($g = 1, .., G$) is modeled by a Gaussian mixture distribution defined by:

$$p(\boldsymbol{y}_i | c_i = g; \boldsymbol{\Psi}_g) = \sum_{r=1}^{R_g} \alpha_{gr} \mathcal{N}\left(\boldsymbol{y}_i; \boldsymbol{\mu}_{gr}, \boldsymbol{\Sigma}_{gr}\right), \tag{3.29}$$

where c_i is the discrete-valued variable in $\{1, \ldots, G\}$ representing the class label of the curve described by the feature vector $\boldsymbol{y}_i$,

$$\boldsymbol{\Psi}_g = \Big(\alpha_{g1}, \ldots, \alpha_{gR_g}, \boldsymbol{\mu}_{g1}, \ldots, \boldsymbol{\mu}_{gR_g}, \ldots, \boldsymbol{\Sigma}_{g1}, \ldots, \boldsymbol{\Sigma}_{gR_g}\Big)$$

is the parameter vector of the mixture of R_g component densities of class g and the corresponding non negative mixing proportions α_{gr} $(r = 1, \ldots, R_g)$ satisfying $\sum_{r=1}^{R_g} \alpha_{gr} = 1$ represent the prior probability of sub-class r $(r = 1, \ldots, R_g)$ for class g $(g = 1, \ldots, G)$. The optimal number of Gaussian distributions R_g for each class g is computed by maximizing the BIC criterion (Schwarz, 1978):

$$\mathrm{BIC}(R_g) = \mathcal{L}(\hat{\boldsymbol{\Psi}}_g) - \frac{\nu_{R_g}}{2} \log(n_g), \tag{3.30}$$

where $\hat{\boldsymbol{\Psi}}_g$ is the maximum likelihood estimate of $\boldsymbol{\Psi}_g$ provided by the EM algorithm, $\nu_{R_g} = \frac{R_g(\nu_{\boldsymbol{\theta}}+1)(\nu_{\boldsymbol{\theta}}+2)}{2} - 1$ is the dimension of the parameter vector $\boldsymbol{\Psi}_g$, and n_g is the cardinal number of class g.

Curve classification by the MAP rule

After performing the supervised learning procedure for MDA using the EM algorithm, one obtains the classes' parameter vectors $(\hat{\boldsymbol{\Psi}}_1, \ldots, \hat{\boldsymbol{\Psi}}_G)$. To classify a new acquired curve, a feature vector is first extracted from this curve by applying the proposed RHLP model. Then, a new curve designed by its feature vector $\boldsymbol{y}_i$, is assigned to the class $\hat{c}_i$ maximizing the posterior probability, that is:

$$\hat{c}_i = \arg \max_{1 \leq g \leq G} p(c_i = g | \boldsymbol{y}_i; \hat{\boldsymbol{\Psi}}_g), \tag{3.31}$$

where

$$p(c_i = g | \boldsymbol{y}_i; \hat{\boldsymbol{\Psi}}_g) = \frac{w_g \; p(\boldsymbol{y}_i | c_i = g; \hat{\boldsymbol{\Psi}}_g)}{\sum_{h=1}^{G} w_h \; p(\boldsymbol{y}_i | c_i = h; \hat{\boldsymbol{\Psi}}_h)}, \tag{3.32}$$

where $w_g = p(c_i = g)$ is the prior probability of class g $(g = 1, \ldots, G)$ calculated as the proportion of the class g in the training set and $p(\boldsymbol{y}_i | c_i = g; \hat{\boldsymbol{\Psi}}_g)$ is the class conditional mixture density for class g given by (3.29) and computed with the estimated parameter vector $\hat{\boldsymbol{\Psi}}_g$.

3.5 Illustration

This paragraph is devoted to an illustration of the curve modeling performed by the proposed RHLP model using nonlinear noisy functions.

The first curve example shown in Figure 3.5 consists in a curve including smooth regime transitions over time, and the second one (c.f., Figure 3.6) includes both smooth and abrupt regime changes. The estimated curve obtained with the RHLP for these two situations clearly shows that RHLP model provides an accurate representation for curves including abrupt and/or smooth transitions. The temporal location and the fuzziness of the regime transitions are controlled by the logistic process. Indeed, as it can be seen on Figure 3.5 (c) and Figure 3.6 (c), the logistic probabilities π_k corresponding to the kth polynomial component are very close to 1 when the kth regime is active, and are closed to 0 otherwise. Figure 3.7 includes solutions provided by alternative regression models, including splines, piecewise regression and HMM regression. One can see that the spline model is not well adapted to handle the problem of abrupt regime changes. On the other hand, the HMMR and the piecewise regression models are more adapted for abrupt regime changes as it can be seen on the bottom plots in Figure 3.7. While, the RHLP model can handle both the abrupt and smooth transitions thanks to the flexibility of the logistic process.

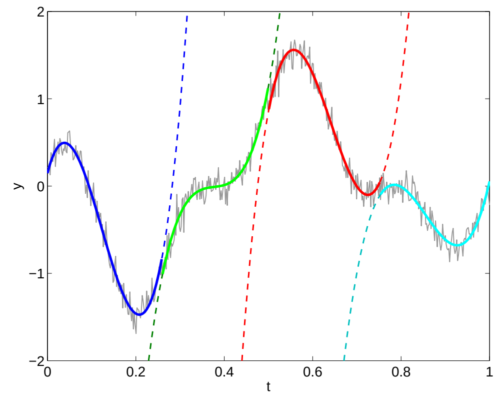
3.6 Summary

In this chapter, we proposed a new approach for curve modeling based on a regression model that incorporates a discrete hidden logistic process. The logistic probability function used for the hidden variables allows for smooth and/or abrupt transitions between various polynomial regression components over time. In addition to curve modeling, the proposed model can provide accurate curve segmentation and smoothing. The EM algorithm provides an elegant framework for estimating the model parameters and the M-step of the EM algorithm uses a multi-class IRLS algorithm to estimate the hidden process parameters.

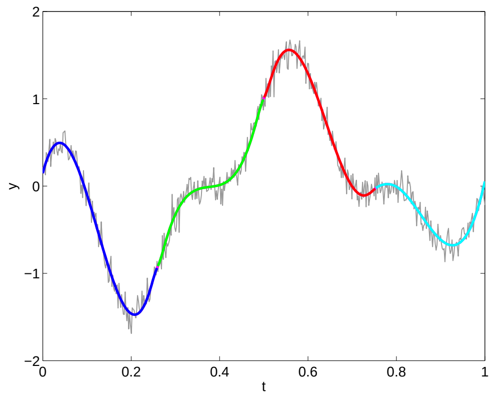
Based on the proposed modeling approach, one can perform curve classification by means of a two-step process, that is, feature extraction from the curves using the RHLP model followed by multidimensional data clustering using the mixture model-based clustering approach or classification using MDA.

The good performance of the proposed modeling approach as compared to other alternative approaches is demonstrated in Chapter 4. In that chapter, we extend the RHLP approach to simultaneously model a set of curves, and a new curve classification rule is then derived.

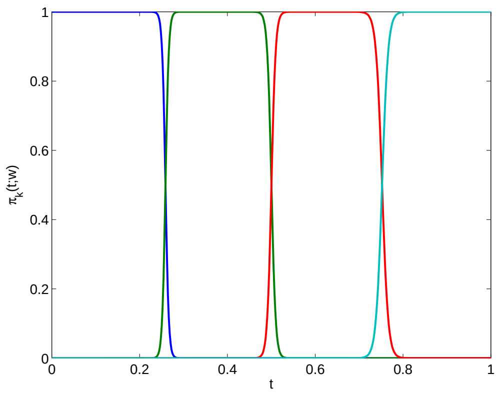
The application to real-world curves from the railway switch operations is reported in Chapter 6.



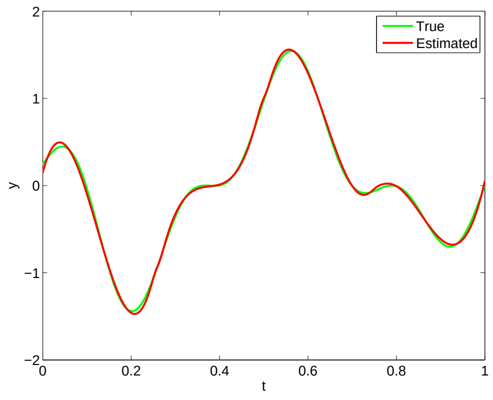
(a)



(b)

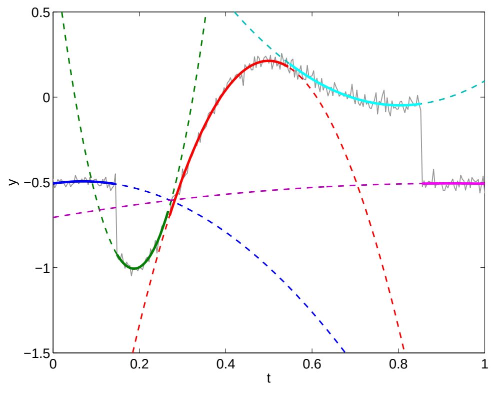


(c)

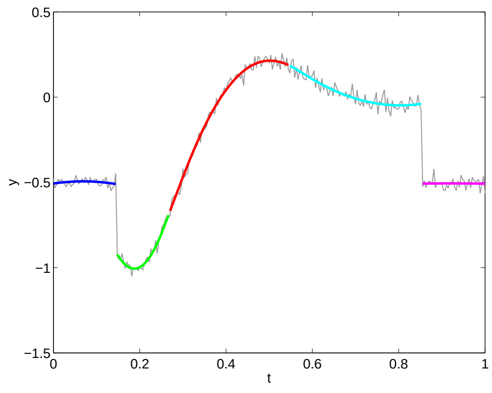


(d)

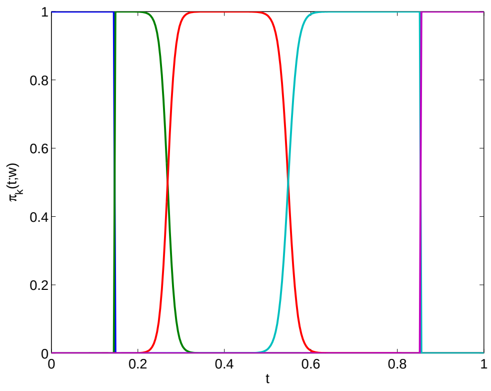
Figure 3.5: Results obtained by applying the RHLP model on a simulated curve of $m = 300$ points where each point is given by: $y_t = e^{-0.4t}(\sin(\pi t) - \sin(6\pi t))\sin(-2.5\pi t) + \epsilon_t$ with $\epsilon_t \sim \mathcal{N}(0, 0.1^2)$. The plot in (a) shows the original noisy curve (in gray) and the estimated polynomial components over time with $K = 4$ and $p = 3$. The polynomial components for which the logistic probabilities are more than 0.5 are shown in continuous line (that is the temporal phase for which the polynomial component is active) and those for which the corresponding logistic proportions are less that 0.5 are shown in dashed line. The logistic probabilities corresponding to the five polynomial components are shown in (c). The plot (b) shows the polynomial components weighted by the logistic probabilities. In (d) we can see the true simulated mean curve and the curve estimated by the proposed approach.



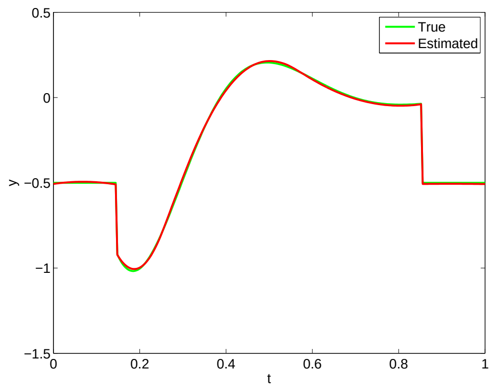
(a)



(b)



(c)



(d)

Figure 3.6: Results obtained by applying the RHLP model on a simulated curve of $m = 300$ points. The curve consists of three piecewise functions where the first and the last are constant (-0.5) and are composed of 40 points, and each point of the second is given by: $y_t = -2e^{-4t}\sin(2.5\pi t) + \epsilon_t$ with $\epsilon_t \sim \mathcal{N}(0, 0.02^2)$. The plot in (a) shows the original noisy curve (in gray) and the estimated polynomial components over time with $K = 5$ and $p = 2$. The polynomial components for which the logistic probabilities are more than 0.5 are shown in continuous line and those for which the corresponding logistic proportions are less that 0.5 are shown in dashed line. The logistic probabilities corresponding to the five polynomial components are shown in (c). The plot (b) shows the polynomial components weighted by the logistic probabilities. In (d) we can see the true simulated mean curve and the curve estimated by the proposed approach.

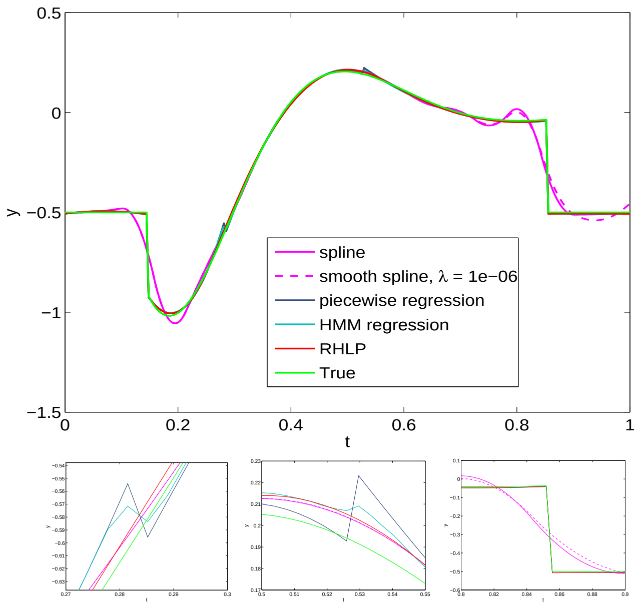


Figure 3.7: The true mean curve corresponding to the second situation presented above and estimated curves obtained by applying different regression models seen in Chapter 3. The splines are quadratic ($p = 2$) with 10 internal knots placed uniformly on the range of t. The HMMR and the RHLP models are used with $K = 5$ and $p = 2$. Each of the three bottom plots presents a zoom for upper plot in a specified location.

Chapter 4

Proposed approach for set of curves modeling and classification

Contents

4.1 Introduction

The work presented in this chapter involves modeling and classifying sets of curves with regime changes. Figure 4.1 shows examples of two sets of curves from the real-world switch operations. In the previous chapter, we saw that the classification task can

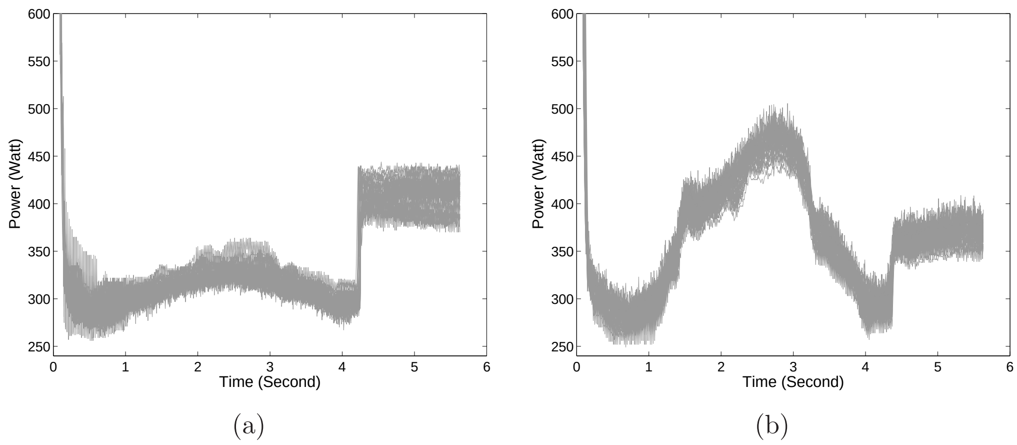


Figure 4.1: Examples of curves of electrical power consumed during various switch operations; 35 curves correspond to operations without defect (a), and 45 curves correspond to operations with critical defect (b).

be achieved by summarizing each curve in a low-dimensional feature vector using the RHLP model and then applying a multidimensional data classification approach.

In this chapter, we extend the regression model with a hidden logistic process (RHLP) presented in the previous chapter for a single curve to a set of curves. The RHLP model applied to a set of curves provides a compact and simplified representation for the curves. In addition, the RHLP model with this specification can be used to directly perform curve classification through the Maximum a Posteriori (MAP) rule (Chamroukhi et al., 2010) and curve clustering. Indeed, the term "direct" refers to the fact that in contrast to the two-fold classification approach found in the previous chapter (i.e., the RHLP model followed by classification or clustering), the approach presented here does not require an external classification model. The curve classification based on the RHLP model for a set of curves is directly performed in the space of curves.

In contrast to classical model-based functional discriminant and functional clustering approaches, the approach presented here uses an RHLP model for each class of curves so that it is particularly appropriate for capturing the dynamical aspect within each class of curves through the underlying logistic process. Both the standard Functional Linear Discriminant Analysis (FLDA) (James and Hastie, 2001) and the regression mixture model (Gaffney, 2004; James and Sugar, 2003) use splines to model curve classes, which require the setting of knots. In addition, we saw in the illustrative example in Chapter 3 that when the curve to be modeled has complex regime transitions, the RHLP model is more promising with respect to providing an accurate approximation. Another alternative approach consists of using the piecewise polynomial regression model that allows for fitting several (polynomial) models to the curves for different time ranges.

We then provide another extension of the RHLP model using an original framework for learning from a set of heterogeneous curves with missing information. This extension leads us to a mixture of RHLP models, abbreviated as MixRHLP, which is particularly appropriate for the supervised classification of curves in the case of complex shaped classes and for curve clustering.

A related method for curve clustering is the segmentation-clustering approach of Picard et al. (2007) applied to array CGH data. In particular, one can also distinguish the recently proposed K-means-like algorithm (Hébrail et al., 2010; Hugueney et al., 2009) for curve clustering and segmentation using piecewise regression (see section 2.7.2).

In addition to the use of the MixRHLP model as model-based curve clustering, we formulate it in a supervised curve classification framework to address the problem of complex shaped classes. The resulting classification approach can be integrated into a Functional Mixture Discriminant Analysis (FMDA) context as presented by Gui and Li (2003) (see section 2.8.2). The advantages of the FMDA presented here is that it includes a dynamical process for modeling the various regimes governing the set of curves.

The parameters of the models are estimated in a maximum likelihood framework through a dedicated Expectation-Maximization (EM) algorithm. For clustering, we also formulate a CEM algorithm to learn the MixRHLP model in a classification maximum likelihood context.

This chapter is organized as follows. Section 4.2 defines the RHLP model for a set of curves and provides a parameter estimation technique using EM. In this section, we also derive the FLDA approach for curve classification based on the RHLP model. In section 4.3, we develop the MixRHLP model for a set of heterogeneous curves. Section 4.3.5 is concerned with curve clustering using the MixRHLP model, and in section 4.3.6, we use the MixRHLP model for curve classification. Finally, section 4.4 presents an experimental study to assess the performance of the proposed approaches in terms of curve modeling, classification and clustering.

4.2 The RHLP model for a set of curves

This section presents the proposed regression model based on a hidden logistic process for modeling a set of curves. We give the scheme of generating a set of curves according to this model and we derive the parameter estimation. Then we show how to approximate and classify curves using this model.

4.2.1 Model definition

In the proposed model, each curve $\mathbf{y}_i$ $(i = 1, \ldots, n)$ is assumed to be generated by the following regression model with a discrete hidden process $\mathbf{z} = (z_1, \ldots, z_m)$:

$$y_{ij} = \boldsymbol{\beta}_{z_j}^T \mathbf{t}_j + \sigma_{z_j} \epsilon_{ij}, \quad \epsilon_{ij} \sim \mathcal{N}(0, 1), \tag{4.1}$$

for $j = 1, \ldots, m$, where $z_j \in \{1, \ldots, K\}$ is a hidden discrete variable representing the label of the polynomial regression model generating y_{ij}. This model can be reformulated in a matrix form as

$$\mathbf{y}_i = \sum_{k=1}^{K} \mathbf{Z}_k (\mathbf{X}\boldsymbol{\beta}_k + \sigma_k \boldsymbol{\epsilon}_i), \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_m), \tag{4.2}$$

where $\mathbf{Z}_k$ is the $m \times m$ diagonal matrix whose diagonal elements are $(z_{1k}, \ldots, z_{mk})$, with $z_{jk} = 1$ if $z_j = k$ (i.e., if y_{ij} is generated by the kth regression model) and $z_{jk} = 0$ otherwise, and

$$\mathbf{X} = \begin{bmatrix} 1 & t_1 & t_1^2 & \ldots & t_1^p \\ 1 & t_2 & t_2^2 & \ldots & t_2^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & t_m & t_m^2 & \ldots & t_m^p \end{bmatrix}$$

is the $m \times (p+1)$ regression matrix. The variable $\boldsymbol{\epsilon}_i = (\epsilon_{i1}, \ldots, \epsilon_{im})^T$ is an $m \times 1$ noise vector distributed according to a Gaussian density with zero mean and identity covariance matrix.
A graphical representation for the RHLP model for a set of curves is presented in Figure 4.2.

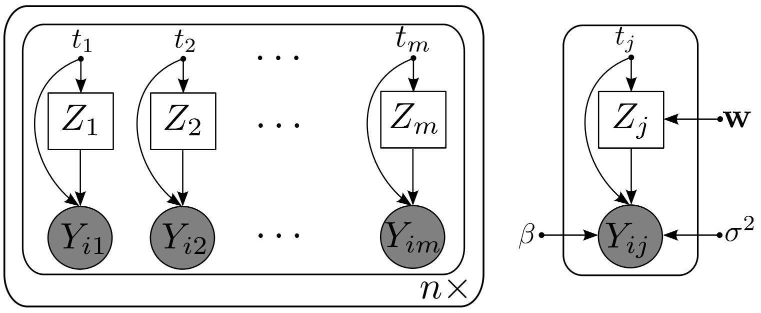


Figure 4.2: Graphical model structure for the proposed regression model with a hidden logistic process for a set of curves (RHLP).

The probability distribution of the process $\mathbf{z} = (z_1, \ldots, z_m)$ that allows for the switching from one regression model to another is defined in the following way. The proposed hidden logistic process assumes that the variables z_j, given the vector $\mathbf{t} = (t_1, \ldots, t_m)$, are generated independently according to the multinomial distribution $\mathcal{M}(1, \pi_1(t_j; \mathbf{w}), \ldots, \pi_K(t_j; \mathbf{w}))$, where

$$\pi_k(t_j; \mathbf{w}) = p(z_j = k | t_j; \mathbf{w}) = \frac{\exp(w_{k0} + w_{k1} t_j)}{\sum_{\ell=1}^{K} \exp(w_{\ell 0} + w_{\ell 1} t_j)}, \tag{4.3}$$

is the logistic transformation of a linear function of the time point t_j, $\boldsymbol{w}_k = (w_{k0}, w_{k1})^T$ is the 2 dimensional coefficient vector for the kth component of (4.3) and $\mathbf{w} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K)$. Thus, given the vector $\mathbf{t} = (t_1, \ldots, t_m)$, the distribution of a given configuration of the process $\mathbf{z}$ can be written as

$$p(\mathbf{z}|\mathbf{t}; \mathbf{w}) = \prod_{j=1}^{m} \prod_{k=1}^{K} p(z_j = k|t_j; \mathbf{w})^{z_{jk}} = \prod_{j=1}^{m} \prod_{k=1}^{K} \left(\frac{\exp{(w_{k0} + w_{k1} t_j)}}{\sum_{\ell=1}^{K} \exp{(w_{\ell 0} + w_{\ell 1} t_j)}} \right)^{z_{jk}}. \quad (4.4)$$

In this particular regression model, the variable z_j controls the switching from one regression model to another of K regression models within the curves at each time t_j.

The use of the logistic process for modeling the sequence of variables z_j allows for modeling both abrupt and smooth regime transitions within the curves, unlike the piecewise regression model, which is adapted only for regimes with abrupt transitions. The relevance of the logistic transformation in terms of flexibility of transitions has been illustrated in Chapter 3 (c.f., section 3.2.1).

4.2.2 Generating a set of curves with the RHLP model

According to the model (4.1), given a regression model k and the time point t_j, the observation y_{ij} is distributed according to a normal density with mean $\boldsymbol{\beta}_k^T \mathbf{t}_j$ and variance σ_k^2, that is:

$$p(y_{ij}|z_j = k, t_j; \boldsymbol{\theta}_k) = \mathcal{N}\big(y_{ij}; \boldsymbol{\beta}_k^T \mathbf{t}_j, \sigma_k^2\big) \quad (4.5)$$

where $\boldsymbol{\theta}_k = (\boldsymbol{\beta}_k, \sigma_k^2)$. Thus, the generative scheme of n curves, given a fixed parameter vector $\boldsymbol{\theta} = (\mathbf{w}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \sigma_1^2, \ldots, \sigma_K^2)$, consists of two steps:

- generate the hidden process $\mathbf{z} = (z_1, \ldots, z_m)$ according to the multinomial distribution $z_j|t_j \sim \mathcal{M}(1, \pi_1(t_j; \mathbf{w}), \ldots, \pi_K(t_j; \mathbf{w}))$, $(j = 1, \ldots, m)$,
- for $i = 1, \ldots, n$ and for $j = 1, \ldots, m$, given $z_j = k$ and t_j, generate each observation y_{ij} according to the Gaussian distribution $\mathcal{N}(\cdot; \boldsymbol{\beta}_k^T \mathbf{t}_j, \sigma_k^2)$.

4.2.3 Parameter estimation by the EM algorithm

From equations (4.1) and (4.5), the distribution of y_{ij} given t_j is the following normal mixture density:

$$\begin{aligned} p(y_{ij}|t_j; \boldsymbol{\theta}) &= \sum_{k=1}^{K} p(z_j = k|t_j; \mathbf{w}) p(y_{ij}|z_j = k, t_j; \boldsymbol{\theta}_k) \\ &= \sum_{k=1}^{K} \pi_k(t_j; \mathbf{w}) \mathcal{N}\big(y_{ij}; \boldsymbol{\beta}_k^T \mathbf{t}_j, \sigma_k^2\big), \quad (4.6) \end{aligned}$$

where $\boldsymbol{\theta} = (\mathbf{w}, \boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_K, \sigma_1^2, \ldots, \sigma_K^2)$ is the parameter vector to be estimated. The parameter $\boldsymbol{\theta}$ is estimated by the maximum likelihood method.

As in the piecewise polynomial regression model, we assume that the curves sample $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ is independent. The independence of the ϵ_{ij}'s $(j = 1, \ldots, m)$ implies the independence of the y_{ij}'s $(j = 1, \ldots, m)$ given the time vector $\mathbf{t} = (t_1, \ldots, t_m)$. It should be noted that the temporal dependence between the underlying segments is controlled by the logistic distribution. The distribution of $\mathbf{y}_i$ given $\mathbf{t}$ is therefore written as

$$p(\mathbf{y}_i|\mathbf{t};\boldsymbol{\theta}) = \prod_{j=1}^{m}\sum_{k=1}^{K}\pi_k(t_j;\mathbf{w})\mathcal{N}\big(y_{ij};\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\big) \tag{4.7}$$

and the log-likelihood of $\boldsymbol{\theta}$ for the observed set of curves $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ at the time points $\mathbf{t}$ is given by:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta};\mathbf{Y},\mathbf{t}) &= \log p(\mathbf{y}_1,\ldots,\mathbf{y}_n|\mathbf{t};\boldsymbol{\theta})\\ &= \log\prod_{i=1}^{n}p(\mathbf{y}_i|\mathbf{t};\boldsymbol{\theta})\\ &= \sum_{i=1}^{n}\sum_{j=1}^{m}\log\sum_{k=1}^{K}\pi_k(t_j;\mathbf{w})\mathcal{N}\big(y_{ij};\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\big).\end{aligned} \tag{4.8}$$

Since this log-likelihood cannot be maximized directly, we use a dedicated Expectation-Maximization (EM) algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997) to perform the maximization.

With this specification, the complete-data log-likelihood of $\boldsymbol{\theta}$ given a configuration $\mathbf{z}$ of the hidden logistic process is written as

$$\begin{aligned}\mathcal{L}_c(\boldsymbol{\theta};\mathbf{Y},\mathbf{z},\mathbf{t}) &= \log p(\mathbf{y}_1,\ldots,\mathbf{y}_n,\mathbf{z}|\mathbf{t};\boldsymbol{\theta})\\ &= \log\prod_{i=1}^{n}p(\mathbf{y}_i,\mathbf{z}|\mathbf{t};\boldsymbol{\theta})\\ &= \log\prod_{i=1}^{n}p(\mathbf{z}|\mathbf{t};\mathbf{w})p(\mathbf{y}_i|\mathbf{z},\mathbf{t};\boldsymbol{\theta})\\ &= \log\prod_{i=1}^{n}\prod_{j=1}^{m}\prod_{k=1}^{K}\left[p(z_j=k|t_j;\mathbf{w})p(y_{ij}|z_j=k,t_j;\boldsymbol{\theta}_k)\right]^{z_{jk}}\\ &= \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{K}z_{jk}\log\left[\pi_k(t_j;\mathbf{w})\mathcal{N}\left(y_{ij};\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\right)\right].\end{aligned} \tag{4.9}$$

The next section gives the proposed EM algorithm for maximizing the observed-data log-likelihood (4.8).

4.2.4 The dedicated EM algorithm

The proposed EM algorithm starts with an initial parameter $\boldsymbol{\theta}^{(0)}$ and alternates between the two following steps until convergence:

E-step

This step computes the expectation of the complete-data log-likelihood (4.9), conditionally on the observed data $\mathbf{Y}$ and the time vector $\mathbf{t}$ using the current parameter estimation denoted by $\boldsymbol{\theta}^{(q)}$ (q being the current iteration):

$$\begin{aligned}
Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)}) &= \mathbb{E}\Big[\mathcal{L}_c(\boldsymbol{\theta};\mathbf{Y},\mathbf{z})|\mathbf{Y},\mathbf{t};\boldsymbol{\theta}^{(q)}\Big] \\
&= \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{K}\mathbb{E}\big[z_{jk}|y_{ij},t_j;\boldsymbol{\theta}^{(q)}\big]\log\big[\pi_k(t_j;\mathbf{w})\mathcal{N}(y_{ij};\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2)\big] \\
&= \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{K}\tau_{ijk}^{(q)}\log\big[\pi_k(t_j;\mathbf{w})\mathcal{N}\left(y_{ij};\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\right)\big] \\
&= \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{K}\tau_{ijk}^{(q)}\log\pi_k(t_j;\mathbf{w})+\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{K}\tau_{ijk}^{(q)}\log\mathcal{N}\left(y_{ij};\boldsymbol{\beta}_k^T\mathbf{t}_j,\sigma_k^2\right), \qquad (4.10)
\end{aligned}$$

where

$$\tau_{ijk}^{(q)} = p(z_{jk}=1|y_{ij},t_j;\boldsymbol{\theta}^{(q)}) = \frac{\pi_{jk}(\mathbf{w}^{(q)})\mathcal{N}(y_{ij};\boldsymbol{\beta}_k^{T(q)}\mathbf{t}_j,\sigma_k^{2(q)})}{\sum_{\ell=1}^{K}\pi_\ell(t_j;\mathbf{w}^{(q)})\mathcal{N}(y_{ij};\boldsymbol{\beta}_\ell^{T(q)}\mathbf{t}_j,\sigma_\ell^{2(q)})} \qquad (4.11)$$

is the posterior probability that y_{ij} originates from the kth regression model.
As shown in the expression of $Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})$, this step simply requires the computation of the posterior probabilities $\tau_{ijk}^{(q)}$.

M-step

In this step, the value of the parameter $\boldsymbol{\theta}$ is updated by maximizing the expected complete-data log-likelihood $Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)})$ with respect to $\boldsymbol{\theta}$, that is:

$$\boldsymbol{\theta}^{(q+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)}). \qquad (4.12)$$

Let $Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})$ and $Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta}_k,\boldsymbol{\theta}^{(q)})$ denote the terms in Equation (4.10) that are function of $\mathbf{w}$ and $\boldsymbol{\theta}_k$ respectively, we have:

$$Q(\boldsymbol{\theta},\boldsymbol{\theta}^{(q)}) = Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)}) + \sum_{k=1}^{K} Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta}_k,\boldsymbol{\theta}^{(q)}), \qquad (4.13)$$

with

$$Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)}) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{K}\tau_{ijk}^{(q)}\log\pi_k(t_j;\mathbf{w}) \qquad (4.14)$$

and, for $k=1,\ldots,K$,

$$\begin{aligned}
Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta}_k,\boldsymbol{\theta}^{(q)}) &= \sum_{i=1}^{n}\sum_{j=1}^{m}\tau_{ijk}^{(q)}\log\mathcal{N}\left(y_{ij};\boldsymbol{\beta}_k^T\mathbf{t}_i,\sigma_k^2\right) \\
&= -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m}\tau_{ijk}^{(q)}\Bigg[\frac{\left(y_{ij}-\boldsymbol{\beta}_k^T\mathbf{t}_j\right)^2}{\sigma_k^2}+\log\sigma_k^2+\log 2\pi\Bigg]. \qquad (4.15)
\end{aligned}$$

Thus, the maximization of $Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{(q)})$ can be performed by separately maximizing $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$ with respect to $\mathbf{w}$ and $Q_{\boldsymbol{\theta}_k}(\boldsymbol{\theta}_k, \boldsymbol{\theta}^{(q)})$ with respect to $\boldsymbol{\theta}_k$ for $k = 1, \ldots, K$.

Consider the update for the regression coefficients $\boldsymbol{\beta}_k$. As it can be seen from Equation (4.15), the regression coefficients update $\boldsymbol{\beta}_k^{(q+1)}$ are obtained by:

$$\boldsymbol{\beta}_k^{(q+1)} = \arg \min_{\boldsymbol{\beta}_k \in \mathbb{R}^{p+1}} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} \tau_{ijk}^{(q)} (y_{ij} - \boldsymbol{\beta}_k^T \mathbf{t}_j)^2 \tag{4.16}$$

which corresponds to solving a weighted least-squares problem. This problem can be solved analytically. Calculating the derivative of the weighted least-squares criterion in (4.16) w.r.t $\boldsymbol{\beta}_k$ and setting it to zero yields:

$$\boldsymbol{\beta}_k^{(q+1)} = \Big[\sum_{i=1}^{n} \sum_{j=1}^{m} \tau_{ijk}^{(q)} \mathbf{t}_j \mathbf{t}_j^T\Big]^{-1} \sum_{i=1}^{n} \sum_{j=1}^{m} \tau_{ijk}^{(q)} y_{ij} \mathbf{t}_j \tag{4.17}$$

which can be written in a matrix form as

$$\boldsymbol{\beta}_k^{(q+1)} = (\mathbf{X}^{*T} \mathbf{W}_k^{(q)} \mathbf{X}^*)^{-1} \mathbf{X}^{*T} \mathbf{W}_k^{(q)} \mathbf{y}^* \tag{4.18}$$

where $\mathbf{W}_k^{(q)}$ is the $nm \times nm$ diagonal matrix whose diagonal elements are the posterior probabilities $(\tau_{11k}^{(q)}, \ldots, \tau_{1mk}^{(q)}, \ldots, \tau_{n1k}^{(q)}, \ldots, \tau_{nmk}^{(q)})$ for the kth regression component, and $\mathbf{X}^* = (\mathbf{X}^T, \ldots, \mathbf{X}^T)^T$ is the $nm \times (p+1)$ matrix composed of n copies of the regression matrix $\mathbf{X}$ and $\mathbf{y}^* = (\mathbf{y}_1^T, \ldots, \mathbf{y}_n^T)^T$ is an $nm \times 1$ column vector composed of all the curves, one curve after another, by stacking the column vectors of $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ using the *vec* operator $\mathbf{y}^* = \text{vec}(\mathbf{Y})$.

Now consider the update for the variance σ_k^2 $(k = 1, \ldots, K)$. The updating value $\sigma_k^{2(q+1)}$ of the variance is obtained by maximizing the terms in Equation (4.15) that depend on σ_k^2, or by equivalence as

$$\sigma_k^{2(q+1)} = \arg \min_{\sigma_k^2 \in \mathbb{R}^+} \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{m} \tau_{ijk}^{(q)} \Big[\frac{(y_{ij} - \boldsymbol{\beta}_k^{T(q+1)} \mathbf{t}_j)^2}{\sigma_k^2} + \log \sigma_k^2\Big] \tag{4.19}$$

which corresponds to a weighted variant of the problem of estimating the variance for an univariate Gaussian density, its closed form solution is given by:

$$\sigma_k^{2(q+1)} = \frac{1}{n m_k^{(q)}} \sum_{i=1}^{n} \sum_{j=1}^{m} \tau_{ijk}^{(q)} (y_{ij} - \boldsymbol{\beta}_k^{T(q+1)} \mathbf{t}_j)^2. \tag{4.20}$$

This can be formulated in a matrix form as

$$\sigma_k^{2(q+1)} = \frac{1}{n m_k^{(q)}} (\mathbf{y}^* - \mathbf{X}^* \boldsymbol{\beta}_k^{(q+1)})^T \mathbf{W}_k^{(q)} (\mathbf{y}^* - \mathbf{X}^* \boldsymbol{\beta}_k^{(q+1)}) \tag{4.21}$$

where $m_k^{(q)} = \sum_{j=1}^{m} \tau_{ijk}^{(q)}$ can be seen as the number of elements in the kth polynomial component for each curve $\mathbf{y}_i$, estimated at iteration q.

Finally, the maximization of $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$ with respect to the parameter vector $\mathbf{w}$ is a multinomial logistic regression problem weighted by the posterior probabilities $\tau_{ijk}^{(q)}$ which can be solved using a multi-class Iterative Reweighted Least Squares (IRLS) algorithm (Chamroukhi et al., 2009c; Chen et al., 1999; Green, 1984; Krishnapuram et al., 2005). The IRLS algorithm is given in the next paragraph.

The Iteratively Reweighted Least Squares (IRLS) algorithm

The IRLS algorithm is used to maximize $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$ with respect to $\mathbf{w} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K)$ in the M-step at each iteration q of the EM algorithm. The vector $\boldsymbol{w}_K$ is set to the null vector to guarantee $\sum_{k=1}^{K} \pi_k(t_j; \mathbf{w}) = 1$. After starting with an initial solution $\mathbf{w}^{(0)}$, the estimation of $\mathbf{w}$ is updated iteratively by the IRLS algorithm where a single update at iteration l is given by

$$\mathbf{w}^{(l+1)} = \mathbf{w}^{(l)} - \left[\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \mathbf{w} \partial \mathbf{w}^T}\right]^{-1}_{\mathbf{w}=\mathbf{w}^{(l)}} \frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \mathbf{w}}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}}. \tag{4.22}$$

The Hessian matrix and the gradient vector of $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$ evaluated at $\mathbf{w} = \mathbf{w}^{(l)}$ are computed similarly as in equations (A.26) and (A.22) respectively, as follows:

$$\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_k \partial \boldsymbol{w}_\ell^T}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} = -\sum_{i=1}^{n}\sum_{j=1}^{m} \pi_k(t_j; \mathbf{w}^{(l)})\big(\delta_{k\ell} - \pi_\ell(t_j; \mathbf{w}^{(l)})\big)\boldsymbol{v}_j \boldsymbol{v}_j^T, \tag{4.23}$$

where $\boldsymbol{v}_j = (1, t_j)^T$. The gradient vector $\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \mathbf{w}}$ is composed of $K-1$ gradient component vectors has the following form:

$$\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \mathbf{w}} = \left(\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_1}, \ldots, \frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_{K-1}}\right)^T \tag{4.24}$$

where each of the $K-1$ gradient component vectors is given by

$$\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_k}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} = \sum_{i=1}^{n}\sum_{j=1}^{m}\big(\tau_{ijk}^{(q)} - \pi_k(t_j; \mathbf{w}^{(l)})\big)\boldsymbol{v}_j. \tag{4.25}$$

After applying the IRLS algorithm (c.f., Equation (4.22)) in the inner loop of the EM algorithm, we obtain the parameter update $\mathbf{w}^{(q+1)}$.

The proposed algorithm is performed with a time complexity of $\mathcal{O}(I_{\text{EM}} I_{\text{IRLS}} nmK^3p^2)$, where I_{EM} is the number of iterations of the EM algorithm and I_{IRLS} is the average number of iterations required by the IRLS algorithm used in the maximization step at each iteration of the EM algorithm. In practice, a particular strategy is used to initialize the IRLS algorithm which reduces its running time. It consists of choosing an arbitrary value of the parameter $\mathbf{w}$ only for the first EM iteration, and then from the second M iteration, the internal IRLS algorithm is initialized with the previous EM estimation of $\mathbf{w}$. In this framework of the EM algorithm, since at each iteration we maximize the Q-function, and according to Jensen's inequality (Jensen, 1906), the observed-data log-likelihood always monotonically increases from one EM iteration to another. In addition, the EM sequence of parameter estimates $\{\boldsymbol{\theta}^{(q)}\}$ is guaranteed to converge toward a local maximum of the log-likelihood function. For further details on the convergence proof of the EM algorithm the reader is referred to the articles of Wu (1983) and Xu and Jordan (1996).

Practical considerations

Since $\mathbf{W}_k$ is an $nm \times nm$ diagonal matrix, direct operations with this matrix in equations (4.18) and (4.21) may be very inefficient. The following modified formulas are provided for efficient calculations:

$$\boldsymbol{\beta}_k^{(q+1)} = (\widetilde{\mathbf{X}}^{*T}\widetilde{\mathbf{X}}^{*})^{-1}\widetilde{\mathbf{X}}^{*T}\widetilde{\mathbf{y}}^{*} \tag{4.26}$$

$$\sigma_k^{2(q+1)} = \frac{1}{nm_k^{(q)}}(\widetilde{\mathbf{y}}^{*} - \widetilde{\mathbf{X}}^{*}\boldsymbol{\beta}_k^{(q+1)})^T(\widetilde{\mathbf{y}}^{*} - \widetilde{\mathbf{X}}^{*}\boldsymbol{\beta}_k^{(q+1)}) \tag{4.27}$$

where $\widetilde{\mathbf{X}}^{*}$ is the $nm \times (p+1)$ matrix, each of its columns is the entry-wise multiplication (Hadamard product, c.f., Appendix B.4) of each column of the matrix $\mathbf{X}^{*}$ by the $nm \times 1$ column vector of the square roots of the posterior probabilities $\boldsymbol{\tau}_k$

$$\boldsymbol{\tau}_k = \left(\sqrt{\tau_{11k}^{(q)}}, \ldots, \sqrt{\tau_{1mk}^{(q)}}, \ldots, \sqrt{\tau_{n1k}^{(q)}}, \ldots, \sqrt{\tau_{nmk}^{(q)}}\right)^T.$$

The $nm \times 1$ column vector $\widetilde{\mathbf{y}}^{*}$ is the Hadamard product of $\mathbf{y}^{*}$ and $\boldsymbol{\tau}_k$.

The pseudo code 4 summarizes the EM algorithm for the proposed RHLP model.

Algorithm 4 Pseudo code of the proposed algorithm for the RHLP model for a set of curves.

Inputs: training set of n curves $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ sampled at the time points $\mathbf{t} = (t_1, \ldots, t_m)$, the number of polynomial components K and the polynomial degree p

```
1:  Initialize: $\boldsymbol{\theta}^{(0)} = (\mathbf{w}^{(0)}, \boldsymbol{\beta}_1^{(0)}, \ldots, \boldsymbol{\beta}_K^{(0)}, \sigma_1^{2(0)}, \ldots, \sigma_K^{2(0)})$
2:  fix a threshold $\epsilon > 0$
3:  set $q \leftarrow 0$ (EM iteration)
4:  while increment in log-likelihood $> \epsilon$ do
5:      E-Step:
6:      for $k = 1, \ldots, K$ do
7:          compute $\tau_{ijk}^{(q)}$ for $i = 1, \ldots, n$ and $j = 1, \ldots, m$ using Equation (4.11)
8:      end for
9:      M-Step:
10:     for for $k = 1, \ldots, K$ do
11:         compute $\boldsymbol{\beta}_k^{(q+1)}$ using Equation (4.26)
12:         compute $\sigma_k^{2(q+1)}$ using Equation (4.27)
13:     end for
14:     IRLS:
15:     Initialize: set $\mathbf{w}^{(l)} = \mathbf{w}^{(q)}$
16:     set a threshold $\delta > 0$
17:     $l \leftarrow 0$ (IRLS iteration)
18:     while increment in $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)}) > \delta$ do
19:         compute $\mathbf{w}^{(l+1)}$ using Equation (4.22)
20:         $l \leftarrow l + 1$
21:     end while
22:     $\mathbf{w}^{(q+1)} \leftarrow \mathbf{w}^{(l)}$
23:     $q \leftarrow q + 1$
24: end while
25: $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(q)}$
```

Output: $\hat{\boldsymbol{\theta}}$ maximum likelihood estimate of $\boldsymbol{\theta}$

4.2.5 Model selection

The optimal value of the pair (K, p) can be computed by using the Bayesian Information Criterion (BIC) (Schwarz, 1978) which is a penalized likelihood criterion, defined by

$$\mathrm{BIC}(K, p) = \mathcal{L}(\hat{\boldsymbol{\theta}}; \mathbf{Y}) - \frac{\nu_{\boldsymbol{\theta}} \log(n)}{2}, \tag{4.28}$$

where $\nu_{\boldsymbol{\theta}} = K(p + 4) - 2$ is the number of free parameters of the model, $\mathcal{L}(\hat{\boldsymbol{\theta}})$ is incomplete-data log-likelihood obtained at convergence of the EM algorithm and n is the sample size. Note that here the sample size refers to the number of curves where each curve has m observations, by equivalence to the case of multidimensional data where the sample comprises n individuals, each of them has d observations (e.g., see the expression of BIC in the case of a set of curves in Liu and Yang (2009)).

4.2.6 Curves approximation and classification with the RHLP model

Approximating a set of curves with the RHLP model

Using the proposed model, the set of curves belonging to the same class is approximated by a single "mean" curve. Each point of this curve is defined by the conditional expectation $\mathbb{E}[y_{ij}|t_j;\hat{\boldsymbol{\theta}}]$, where

$$\begin{aligned}\mathbb{E}[y_{ij}|t_j;\hat{\boldsymbol{\theta}}] &= \int_{\mathbb{R}} y_{ij}p(y_{ij}|t_j;\hat{\boldsymbol{\theta}})dy_{ij}\\ &= \sum_{k=1}^{K}\pi_k(t_j;\hat{\mathbf{w}})\int_{\mathbb{R}} y_{ij}\mathcal{N}\big(y_{ij};\hat{\boldsymbol{\beta}}_k^T\mathbf{t}_j,\hat{\sigma}_k^2\big)dy_{ij}\\ &= \sum_{k=1}^{K}\pi_k(t_j;\hat{\mathbf{w}})\hat{\boldsymbol{\beta}}_k^T\mathbf{t}_j,\end{aligned} \tag{4.29}$$

$\hat{\boldsymbol{\theta}} = (\hat{\mathbf{w}},\hat{\boldsymbol{\beta}}_1,\ldots,\hat{\boldsymbol{\beta}}_K,\hat{\sigma}_1^2,\ldots,\hat{\sigma}_K^2)$ being the parameter vector obtained at convergence of the algorithm. The matrix formulation of the mean curve is given by:

$$\hat{\mathbf{y}} = \sum_{k=1}^{K}\hat{\mathbf{\Pi}}_k\mathbf{X}\hat{\boldsymbol{\beta}}_k, \tag{4.30}$$

where $\hat{\mathbf{\Pi}}_k = \mathrm{diag}(\pi_k(t_1;\hat{\mathbf{w}}),\ldots,\pi_k(t_m;\hat{\mathbf{w}}))$ is the $m\times m$ diagonal matrix whose diagonal elements are the logistic proportions associated with the kth regression model.

Curve classification

A curve discrimination rule can be derived from the Bayes theorem, which is similar to the Functional Linear Discriminant Analysis (FLDA) rule proposed by (James and Hastie, 2001). Given a training set of n labeled curves $((\mathbf{y}_1,c_1),\ldots,(\mathbf{y}_n,c_n))$ where c_i represents the class label of the curve $\mathbf{y}_i$ $(i=1,\ldots,n)$, the parameters $(\boldsymbol{\theta}_1,\ldots,\boldsymbol{\theta}_G)$ of the G classes of curves are first estimated by applying the proposed RHLP model to each class of curves. Basing on the learned classes' parameters, the curve classification task is then performed in "functional space". Thus, once the classes parameters are estimated by the EM algorithm, a new acquired curve $\mathbf{y}_i$ is directly assigned to the class $\hat{c}_i$ by the MAP rule:

$$\hat{c}_i = \arg\max_{1\leq g\leq G} p(c_i = g|\mathbf{y}_i,\mathbf{t};\hat{\boldsymbol{\theta}}_g), \tag{4.31}$$

where, for each class g $(g=1,\ldots,G)$,

$$p(c_i = g|\mathbf{y}_i,\mathbf{t};\hat{\boldsymbol{\theta}}_g) = \frac{w_g p(\mathbf{y}_i|c_i=g,\mathbf{t};\hat{\boldsymbol{\theta}}_g)}{\sum_{g'=1}^{G} w_{g'}p(\mathbf{y}_i|c_i=g';\hat{\boldsymbol{\theta}}_{g'})} \tag{4.32}$$

is the posterior probability, $w_g = p(c_i = g)$ is the prior probability estimated by the proportion of the curves belonging to class g in the training phase, $p(\mathbf{y}_i|c_i = g, \mathbf{t}; \boldsymbol{\theta}_g)$ is the conditional density of $\mathbf{y}_i$ defined by Equation (4.7) as

$$p(\mathbf{y}_i|c_i = g, \mathbf{t}; \boldsymbol{\theta}_g) = \prod_{j=1}^{m} \sum_{k=1}^{K} \pi_k(t_j; \mathbf{w}_g) \mathcal{N}\big(y_{ij}; \boldsymbol{\beta}_{gk}^T \mathbf{t}_j, \sigma_{gk}^2\big) \qquad (4.33)$$

and $\hat{\boldsymbol{\theta}}_g$ is the maximum likelihood estimation of $\boldsymbol{\theta}_g = (\mathbf{w}_g, \boldsymbol{\beta}_{g1}, \ldots, \boldsymbol{\beta}_{gK}, \sigma_{g1}^2, \ldots, \sigma_{gK}^2)$.

Behavior of the proposed approach for complex shaped classes

It should be mentioned that this approach may have limitations in the case of an heterogeneous set of curves, as it provides a single model which is more suitable for an homogeneous set of curves. This paragraph illustrates this limitation through a simulated example. Consider simulated curves from a class which is composed two subclasses (see Figure 4.3). As it can be clearly seen in Figure 4.3 (left), the representation provided by the RHLP model is not adapted.

To deal with the problem of complex shaped classes of curves, the next section integrates the RHLP model into a mixture framework. The resulting model is able to handle the problem of heterogeneity in a set of curves. Adopting this mixture approach clearly improves the modeling accuracy, as it can be seen in Figure 4.3 (right), where an adapted representation is provided for each sub-class.

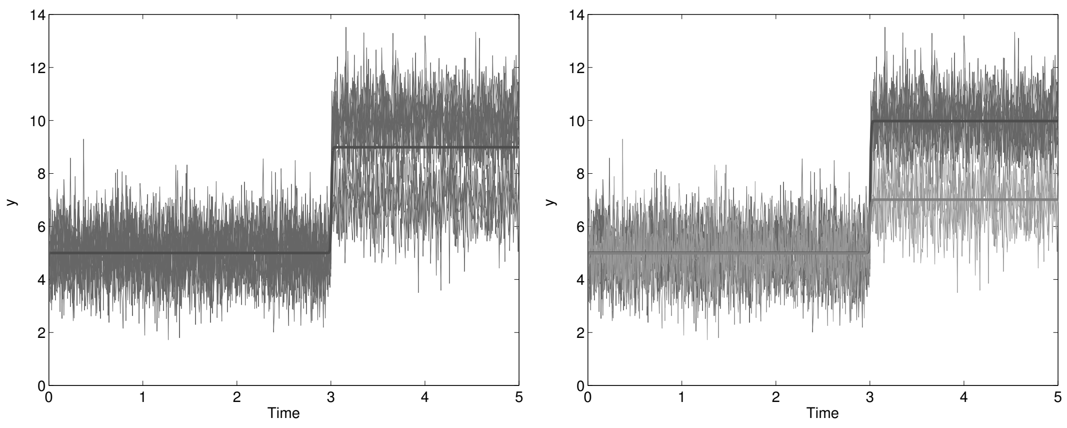


Figure 4.3: Illustrative example of a non homogeneous class having two sub-classes (composed respectively of 10 and 5 curves with $m = 350$ points per curve) generated as piecewise function over time, with the results obtained by the RHLP model (left) and the MixRHLP model (right).

In the next section we develop the mixture of regression models with hidden logistic processes (MixRHLP) for a set of curves and its parameter estimation procedure in a maximum likelihood framework.

4.3 Mixture of regression models with hidden logistic process (MixRHLP)

4.3.1 Model definition

According to the mixture of regression models with hidden logistic processes (MixRHLP), the population of curves $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ is assumed to be made up of R homogeneous subpopulations, mixed at random in proportion to the relative group sizes $\alpha_1, \ldots, \alpha_R$. Each of the R subpopulations of curves is governed by K[1] polynomial regimes and is modeled by the regression model with hidden logistic process (RHLP) presented in section 4.2, that is to say, the distribution of a curve $\mathbf{y}_i$ $(i = 1, \ldots, n)$ given a sub-class $r \in \{1, \ldots, R\}$, is defined as in Equation (4.33) by:

$$p(\mathbf{y}_i|h_i = r, \mathbf{t}; \boldsymbol{\theta}_r) = \prod_{j=1}^{m} \sum_{k=1}^{K} \pi_k(t_j; \mathbf{w}_r) \mathcal{N}\big(y_{ij}; \boldsymbol{\beta}_{rk}^T \mathbf{t}_j, \sigma_{rk}^2\big) \qquad (4.34)$$

where h_i is a discrete-valued variable representing the hidden label of the sub-class generating the curve $\mathbf{y}_i$ $(i = 1, \ldots, n)$ and $\boldsymbol{\theta}_r = (\mathbf{w}_r, \boldsymbol{\beta}_{r1}, \ldots, \boldsymbol{\beta}_{rK}, \sigma_{r1}^2, \ldots, \sigma_{rK}^2)$ is the parameter vector of the sub-class r. Thus, the resulting distribution of each curve $\mathbf{y}_i$ given the time points $\mathbf{t}$ can be defined by the following mixture density:

$$\begin{aligned} p(\mathbf{y}_i|\mathbf{t}; \boldsymbol{\Psi}) &= \sum_{r=1}^{R} p(h_i = r) p(\mathbf{y}_i|h_i = r, \mathbf{t}; \boldsymbol{\theta}_r) \\ &= \sum_{r=1}^{R} \alpha_r \prod_{j=1}^{m} \sum_{k=1}^{K} \pi_k(t_j; \mathbf{w}_r) \mathcal{N}\big(y_{ij}; \boldsymbol{\beta}_{rk}^T \mathbf{t}_j, \sigma_{rk}^2\big) \qquad (4.35) \end{aligned}$$

that is the MixRHLP model. The α_r's $(r = 1, \ldots, R)$ are the non negative mixing proportions that sum to 1 and $\boldsymbol{\Psi} = (\alpha_1, \ldots, \alpha_R, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_R)$ is the parameter vector of the MixRHLP model.

In this specific mixture model, each sub-class (or cluster) of curves r is governed by its own hidden logistic process $\mathbf{z}_r = (z_{r1}, \ldots, z_{rm})$ that allows for the switching from one regression model to another among K polynomial models. The variable z_{rj} $(j = 1, \ldots, m)$ is a discrete-valued variable in $\{1, \ldots, K\}$ such that $z_{rj} = k$ indicates that y_{ij} belongs to the kth polynomial component of sub-class r. The process $\mathbf{z}_r$ is assumed to be logistic and parametrized by the parameter vector $\mathbf{w}_r$, as defined in section 4.4. According to this definition, the variables z_{rj}, given the time vector $\mathbf{t} = (t_1, \ldots, t_m)$, are generated independently according to the multinomial distribution $\mathcal{M}(1, \pi_1(t_j; \mathbf{w}_r), \ldots, \pi_K(t_j; \mathbf{w}_r))$, where the probability distribution of each component k is logistic, that is:

$$\pi_k(t_j; \mathbf{w}_r) = p(z_{rj} = k|t_j; \mathbf{w}_r) = \frac{\exp\left(w_{rk0} + w_{rk1} t_j\right)}{\sum_{\ell=1}^{K} \exp\left(w_{r\ell 0} + w_{r\ell 1} t_j\right)}, \qquad (4.36)$$

[1]Notice that in this section we consider a common value of K for all the model components, however, the model can be easily formulated with a value K_r for each of the R model components.

$\boldsymbol{w}_{rk} = (w_{rk0}, w_{rk1})^T$ being the 2-dimensional coefficient vector for the kth logistic component and $\mathbf{w}_r = (\boldsymbol{w}_{r1}, \ldots, \boldsymbol{w}_{rK})$. The probability distribution of a particular configuration $\mathbf{z}_r$ of the underlying process governing the sub-class r is therefore given by:

$$p(\mathbf{z}_r|\mathbf{t};\mathbf{w}_r) = \prod_{j=1}^{m}\prod_{k=1}^{K}\left(\pi_k(t_j;\mathbf{w}_r)\right)^{z_{rjk}}, \qquad (4.37)$$

where z_{rjk} is a binary variable such that $z_{rjk} = 1$ if $z_{rj} = k$ (i.e., the ith curve belongs to the sub-class r and the jth point y_{ij} of that curve belongs to the kth regime) and $z_{rjk} = 0$ otherwise.

A matrix formulation of the model for a heterogeneous set of curves is formulated as follows. Knowing the labels of sub-groups within the set of n curves, which we denote by the vector $\mathbf{h} = (h_1, \ldots, h_n)$ where $h_i \in \{1, \ldots, R\}$, and a configuration of the logistic process $\mathbf{z}_r$ $(r = 1, \ldots, R)$ governing each of the R sub-classes, the model can therefore be reformulated as

$$\mathbf{y}_i = \sum_{r=1}^{R} h_{ir}\sum_{k=1}^{K}\mathbf{Z}_{rk}(\mathbf{X}\boldsymbol{\beta}_{rk} + \sigma_{rk}\boldsymbol{\epsilon}_i) \quad ; \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_m) \qquad (4.38)$$

where h_{ir} is a binary variable such that $h_{ir} = 1$ if $h_i = r$ (i.e., when $\mathbf{y}_i$ is issued from the sub-class (cluster) r) and $h_{ir} = 0$ otherwise, $\mathbf{Z}_{rk}$ is an $m \times m$ diagonal matrix whose diagonal elements are $(z_{r1k}, \ldots, z_{rmk})$ and $\boldsymbol{\epsilon}_i = (\epsilon_{i1}, \ldots, \epsilon_{im})^T$ is an $m \times 1$ vector of i.i.d standard Gaussian variables.

This model can be understood as follows. The variable h_i refers to the group generating the ith curve $\mathbf{y}_i$. Knowing the group $h_i = r$ of that curve, the variable z_{rj} refers to the regime (polynomial component) generating the jth observation y_{ij}.

4.3.2 Generating a set of curves with the MixRHLP model

The generative model of n curves from a given parameter vector $\boldsymbol{\Psi} = (\alpha_1, \ldots, \alpha_R, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_R)$ consists of the following three steps:

For $i = 1, \ldots, n$:

- generate the hidden variable representing the cluster label according to the multinomial distribution, that is $h_i \sim \mathcal{M}(1, \alpha_1, \ldots, \alpha_R)$

For $j = 1, \ldots, m$

- given the cluster label $h_i = r$, generate the hidden variable representing the polynomial regime at each time point t_j according to the multinomial distribution: $z_{rj}|t_j \sim \mathcal{M}(1, \pi_1(t_j;\mathbf{w}_r), \ldots, \pi_K(t_j;\mathbf{w}_r))$,
- finally, given the cluster label $h_i = r$ and the polynomial component $z_{rj} = k$ the observation y_{ij} is generated according to the Gaussian distribution $\mathcal{N}(\cdot;\boldsymbol{\beta}_{rk}^T\mathbf{t}_j, \sigma_{rk}^2)$.

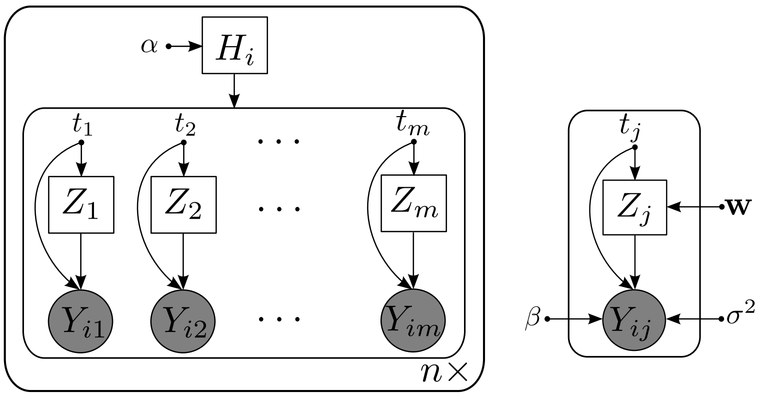


Figure 4.4: Graphical model structure for the mixture of regression models with hidden logistic processes for a set of curves (MixRHLP).

This generative scheme for a set of n curves is illustrated by the graphical representation given in Figure 4.4

The next section presents the unsupervised learning of the model parameters from a set of curves through the EM algorithm.

4.3.3 Maximum likelihood estimation via the EM algorithm

The parameter vector $\mathbf{\Psi}$ of the proposed MixRHLP model is estimated by maximum likelihood from a set of curves. Assuming that the set of n curves $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ are independent and identically distributed, the log-likelihood of $\mathbf{\Psi}$ given the observed data $\mathbf{Y}$ and the sampling time points $\mathbf{t} = (t_1, \ldots, t_m)$ is then written as:

$$\begin{aligned}\mathcal{L}(\mathbf{\Psi}; \mathbf{Y}, \mathbf{t}) &= \log p(\mathbf{y}_1, \ldots, \mathbf{y}_n | \mathbf{t}; \mathbf{\Psi}) = \log \prod_{i=1}^{n} p(\mathbf{y}_i | \mathbf{t}; \mathbf{\Psi}) \\ &= \sum_{i=1}^{n} \log \sum_{r=1}^{R} \alpha_r \prod_{j=1}^{m} \sum_{k=1}^{K} \pi_k(t_j; \mathbf{w}_r) \mathcal{N}\big(y_{ij}; \boldsymbol{\beta}_{rk}^T \mathbf{t}_j, \sigma_{rk}^2\big). \end{aligned} \quad (4.39)$$

The maximization of this log-likelihood cannot be performed in a closed form. We maximize it iteratively by using a dedicated EM algorithm. In this context of the EM algorithm, the complete data consist of the observed set of curves $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ sampled at the time points $\mathbf{t} = (t_1, \ldots, m)$, their corresponding vector of class labels $\mathbf{h} = (h_1, \ldots, h_n)$ and the vector of regime labels $\mathbf{z}_r = (z_{r1}, \ldots, z_{rm})$ for each group r $(r = 1, \ldots, R)$. The complete-data likelihood of $\mathbf{\Psi}$ is therefore given by:

$$\begin{aligned} p(\mathbf{Y}, \mathbf{h}, \mathbf{z}_1, \ldots, \mathbf{z}_R | \mathbf{t}; \mathbf{\Psi}) &= p(\mathbf{h}) p(\mathbf{Y}, \mathbf{z}_1, \ldots, \mathbf{z}_R | \mathbf{h}, \mathbf{t}; \mathbf{\Psi}) \\ &= p(h_1, \ldots, h_n) p(\mathbf{y}_1, \ldots, \mathbf{y}_n, \mathbf{z}_1, \ldots, \mathbf{z}_R | \mathbf{h}, \mathbf{t}; \mathbf{\Psi}) \\ &= \prod_{i=1}^{n} p(h_i) p(\mathbf{z}_{h_i} | \mathbf{t}; \mathbf{w}_{h_i}) p(\mathbf{y}_i | \mathbf{z}_{h_i}, \mathbf{t}; \boldsymbol{\theta}_{h_i}). \end{aligned} \quad (4.40)$$

where in the first step we have used the fact that $\mathbf{h}$ does not depend on $\mathbf{t}$. By using the fact that the class label of a curve h_i is in $\{1,\ldots,R\}$ and plugging the conditional density of a curve (c.f., Equation (3.8)) and the distribution of $\mathbf{z}_r$ (c.f., Equation (4.37)) into the expression of the complete-data likelihood (4.40) leads to

$$\begin{aligned}
p(\mathbf{Y},\mathbf{h},\mathbf{z}_1,\ldots,\mathbf{z}_R|\mathbf{t};\boldsymbol{\Psi}) &= \prod_{i=1}^{n}\prod_{r=1}^{R}\left[p(h_i=r)p(\mathbf{z}_r|\mathbf{t};\mathbf{w}_r)p(\mathbf{y}_i|\mathbf{z}_r,\mathbf{t};\boldsymbol{\theta}_r)\right]^{h_{ir}}\\
&= \prod_{i=1}^{n}\prod_{r=1}^{R}p(h_i=r)^{h_{ir}}\prod_{j=1}^{m}\prod_{k=1}^{K}\left[p(z_{rj}=k|t_j;\mathbf{w}_r)p(y_{ij}|z_{rj}=k,t_j;\boldsymbol{\theta}_{rk})\right]^{z_{rjk}h_{ir}}\\
&= \prod_{i=1}^{n}\prod_{r=1}^{R}\alpha_r^{h_{ir}}\prod_{j=1}^{m}\prod_{k=1}^{K}\left[\pi_k(t_j;\mathbf{w}_r)\mathcal{N}(y_{ij};\boldsymbol{\beta}_{rk}^T t_j,\sigma_{rk}^2)\right]^{z_{rjk}h_{ir}} \qquad (4.41)
\end{aligned}$$

where $\boldsymbol{\theta}_{rk}=(\boldsymbol{\beta}_{rk},\sigma_{rk}^2)$. The complete-data log-likelihood is finally given by taking the logarithm of (4.41):

$$\begin{aligned}
\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{t},\mathbf{h},\mathbf{z}_1,\ldots,\mathbf{z}_R) &= \sum_{i=1}^{n}\sum_{r=1}^{R}h_{ir}\log\alpha_r+\sum_{i=1}^{n}\sum_{r=1}^{R}\sum_{j=1}^{m}\sum_{k=1}^{K}h_{ir}z_{rjk}\log\pi_k(t_j;\mathbf{w}_r)\\
&+\sum_{i=1}^{n}\sum_{r=1}^{R}\sum_{j=1}^{m}\sum_{k=1}^{K}h_{ir}z_{rjk}\log\mathcal{N}\left(y_{ij};\boldsymbol{\beta}_{rk}^T\mathbf{t}_j,\sigma_{rk}^2\right). \qquad (4.42)
\end{aligned}$$

The EM algorithm alternates iteratively between the computation of the conditional expectation of complete-data log-likelihood (4.42) and the maximization of this conditional expectation given a current parameter estimation, until convergence.

The dedicated EM algorithm

The proposed EM algorithm starts with an initial parameter $\boldsymbol{\Psi}^{(0)}$ and alternates between the two following steps until convergence:

E-step: Compute the conditional expectation of the complete-data log-likelihood given the observations $\mathbf{Y}$, the time vector $\mathbf{t}$ and the current parameter estimation $\boldsymbol{\Psi}^{(q)}$:

$$\begin{aligned}
Q(\boldsymbol{\Psi},\boldsymbol{\Psi}^{(q)}) &= \mathbb{E}\left[\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{t},\mathbf{h},\mathbf{z}_1,\ldots,\mathbf{z}_R)|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)}\right]\\
&= \sum_{i=1}^{n}\sum_{r=1}^{R}\mathbb{E}_h\left[h_{ir}|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)}\right]\log\alpha_r+\sum_{i=1}^{n}\sum_{r=1}^{R}\sum_{j=1}^{m}\sum_{k=1}^{K}\mathbb{E}_{h,z}\left[h_{ir}z_{rjk}|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)}\right]\log\pi_k(t_j;\mathbf{w}_r)\\
&+\sum_{i=1}^{n}\sum_{r=1}^{R}\sum_{j=1}^{m}\sum_{k=1}^{K}\mathbb{E}_{h,z}\left[h_{ir}z_{rjk}|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)}\right]\log\mathcal{N}\left(y_{ij};\boldsymbol{\beta}_{rk}^T\mathbf{t}_j,\sigma_{rk}^2\right)\\
&= \sum_{i=1}^{n}\sum_{r=1}^{R}\gamma_{ir}^{(q)}\log\alpha_r+\sum_{i=1}^{n}\sum_{r=1}^{R}\sum_{j=1}^{m}\sum_{k=1}^{K}\gamma_{ir}^{(q)}\tau_{ijrk}^{(q)}\log\pi_k(t_j;\mathbf{w}_r)\\
&+\sum_{i=1}^{n}\sum_{r=1}^{R}\sum_{j=1}^{m}\sum_{k=1}^{K}\gamma_{ir}^{(q)}\tau_{ijrk}^{(q)}\log\mathcal{N}\left(y_{ij};\boldsymbol{\beta}_{rk}^T\mathbf{t}_j,\sigma_{rk}^2\right), \qquad (4.43)
\end{aligned}$$

where the notation $\mathbb{E}_h$ means that the expectation is computed over the variable h indicating the sub-class membership and $\mathbb{E}_{h,z}$ means that the expectation is computed over both the variable h and the variable z indicating polynomial component membership for each sub-class,

$$\begin{aligned}\gamma_{ir}^{(q)} &= p(h_{ir}=1|\mathbf{y}_i,\mathbf{t};\boldsymbol{\Psi}^{(q)})\\ &= \frac{\alpha_r^{(q)}p(\mathbf{y}_i|h_i=r,\mathbf{t};\boldsymbol{\theta}_r^{(q)})}{\sum_{r'=1}^{R}\alpha_{r'}^{(q)}p(\mathbf{y}_i|h_i=r',\mathbf{t};\boldsymbol{\theta}_{r'}^{(q)})}\\ &= \frac{\alpha_r^{(q)}\prod_{j=1}^{m}\sum_{k=1}^{K}\pi_k(t_j;\mathbf{w}_r^{(q)})\mathcal{N}\big(y_{ij};\boldsymbol{\beta}_{rk}^{T(q)}\mathbf{t}_j,\sigma_{rk}^{2(q)}\big)}{\sum_{r'=1}^{R}\alpha_{r'}^{(q)}\prod_{j=1}^{m}\sum_{k=1}^{K}\pi_k(t_j;\mathbf{w}_{r'}^{(q)})\mathcal{N}(y_{ij};\boldsymbol{\beta}_{r'k}^{(q)T}\mathbf{t}_j,\sigma_{r'k}^{2(q)})}\end{aligned} \tag{4.44}$$

is the posterior probability of sub-class r and

$$\begin{aligned}\tau_{ijrk}^{(q)} &= p(z_{rjk}=1|y_{ij},t_j,h_i=r;\boldsymbol{\Psi}^{(q)})=p(z_{rj}=k|y_{ij},t_j;\boldsymbol{\theta}_r^{(q)})\\ &= \frac{\pi_k(t_j;\mathbf{w}_r^{(q)})\mathcal{N}(y_{ij};\boldsymbol{\beta}_{rk}^{T(q)}\mathbf{t}_j,\sigma_{rk}^{2(q)})}{\sum_{\ell=1}^{K}\pi_\ell(t_j;\mathbf{w}_r^{(q)})\mathcal{N}(y_{ij};\boldsymbol{\beta}_{r\ell}^{T(q)}\mathbf{t}_j,\sigma_{r\ell}^{2(q)})}\end{aligned} \tag{4.45}$$

is the posterior probability that the observed data point y_{ij} at time t_j originates from the kth regression model of sub-class r. We note that in computing the conditional expectation $\mathbb{E}_h$ and $\mathbb{E}_{h,z}$ we used the fact that conditional expectations and conditional probabilities are the same for the indicator binary-valued variables h_{ir}, z_{rjk} so that $\mathbb{E}[h_{ir}|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)}]=p(h_{ir}=1|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)})$ and $\mathbb{E}[z_{rjk}|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)}]=p(z_{rjk}=1|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)})$. Additionally, since the set of curves $\mathbf{Y}=(\mathbf{y}_1,\ldots,\mathbf{y}_n)$ is assumed to be independent, we therefore have $p(h_{ir}=1|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)})=p(h_{ir}=1|\mathbf{y}_i,\mathbf{t};\boldsymbol{\Psi}^{(q)})$, and similarly for each curve $\mathbf{y}_i$ we have $p(z_{rjk}=1|\mathbf{y}_i,\mathbf{t};\boldsymbol{\theta}_r^{(q)})=p(z_{rjk}=1|y_{ij},t_j;\boldsymbol{\theta}_r^{(q)})$. The conditional expectation over the variables h and z is obtained in the same manner as

$$\begin{aligned}\mathbb{E}_{h,z}[h_{ir}z_{h_ijk}|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)}] &= p(h_{ir}=1,z_{rjk}=1|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)})\\ &= p(h_i=r|\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)})p(z_{h_ij}=k|h_i=r,\mathbf{Y},\mathbf{t};\boldsymbol{\Psi}^{(q)})\\ &= p(h_i=r|\mathbf{y}_i,\mathbf{t};\boldsymbol{\Psi}^{(q)})p(z_{rj}=k|y_{ij},t_j;\boldsymbol{\Psi}^{(q)})\\ &= \gamma_{ir}^{(q)}\tau_{ijrk}^{(q)}.\end{aligned} \tag{4.46}$$

As shown in the expression of $Q(\boldsymbol{\Psi},\boldsymbol{\Psi}^{(q)})$ (c.f., Equation (4.43)), this step simply requires the computation of the posterior group probabilities $\gamma_{ir}^{(q)}$ and the posterior polynomial regression component probabilities $\tau_{ijrk}^{(q)}$ for K polynomial regimes for each group r $(r=1,\ldots,R)$.

M-step: Update the value of the parameter $\boldsymbol{\Psi}$ by maximizing the function $Q(\boldsymbol{\Psi},\boldsymbol{\Psi}^{(q)})$ with respect to $\boldsymbol{\Psi}$, that is:

$$\boldsymbol{\Psi}^{(q+1)}=\arg\max_{\boldsymbol{\Psi}\in\boldsymbol{\Omega}}Q(\boldsymbol{\Psi},\boldsymbol{\Psi}^{(q)}), \tag{4.47}$$

where $\boldsymbol{\Omega}$ is the parameter space.

To perform this maximization, the Q-function is first decomposed as

$$Q(\mathbf{\Psi}, \mathbf{\Psi}^{(q)}) = Q_\alpha(\alpha_1, \ldots, \alpha_R, \mathbf{\Psi}^{(q)}) + \sum_{r=1}^{R} \big[Q_{\mathbf{w}_r}(\mathbf{w}_r, \mathbf{\Psi}^{(q)}) + \sum_{k=1}^{K} Q_{\boldsymbol{\theta}_{rk}}(\boldsymbol{\theta}_{rk}, \mathbf{\Psi}^{(q)})\big], \quad (4.48)$$

where

$$Q_\alpha(\alpha_1, \ldots, \alpha_R, \mathbf{\Psi}^{(q)}) = \sum_{i=1}^{n}\sum_{r=1}^{R} \gamma_{ir}^{(q)} \log \alpha_r, \quad (4.49)$$

$$Q_{\mathbf{w}_r}(\mathbf{w}_r, \mathbf{\Psi}^{(q)}) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{K} \gamma_{ir}^{(q)} \tau_{ijrk}^{(q)} \log \pi_k(t_j; \mathbf{w}_r) \quad (4.50)$$

and

$$Q_{\boldsymbol{\theta}_{rk}}(\boldsymbol{\theta}_{rk}, \mathbf{\Psi}^{(q)}) = \sum_{i=1}^{n}\sum_{j=1}^{m} \gamma_{ir}^{(q)} \tau_{ijrk}^{(q)} \log \mathcal{N}\left(y_{ij}; \boldsymbol{\beta}_{rk}^T \mathbf{t}_j, \sigma_{rk}^2\right). \quad (4.51)$$

Then, the maximization of $Q(\mathbf{\Psi}, \mathbf{\Psi}^{(q)})$ is performed by the separate maximizations of $Q_\alpha(\alpha_1, \ldots, \alpha_R, \mathbf{\Psi}^{(q)})$, $Q_{\mathbf{w}_r}(\mathbf{w}_r, \mathbf{\Psi}^{(q)})$ and $Q_{\boldsymbol{\theta}_{rk}}(\boldsymbol{\theta}_{rk}, \mathbf{\Psi}^{(q)})$. The function $Q_\alpha(\alpha_1, \ldots, \alpha_R, \mathbf{\Psi}^{(q)})$ is maximized with respect to the mixing proportions $(\alpha_1, \ldots, \alpha_R)$ subject to the constraint $\sum_{r=1}^{R} \alpha_r = 1$. The updates are given by (see Appendix A.1):

$$\alpha_r^{(q+1)} = \frac{1}{n}\sum_{i=1}^{n} \gamma_{ir}^{(q)}, \quad (r = 1, \ldots, R). \quad (4.52)$$

For each group r $(r = 1, \ldots, R)$, the maximization of $Q_{\boldsymbol{\theta}_{rk}}(\boldsymbol{\theta}_{rk}, \mathbf{\Psi}^{(q)})$ with respect to $\boldsymbol{\beta}_{rk}$ $(k = 1, \ldots, K)$ includes separate analytic solutions of K weighted least-squares problems, each of them is defined as

$$\boldsymbol{\beta}_{rk}^{(q+1)} = \arg \min_{\boldsymbol{\beta}_{rk} \in \mathbb{R}^{p+1}} \sum_{i=1}^{n}\sum_{j=1}^{m} \gamma_{ir}^{(q)} \tau_{ijrk}^{(q)} (y_{ij} - {\boldsymbol{\beta}_{rk}}^T \mathbf{t}_j)^2. \quad (4.53)$$

This task is performed in a similar way as for (4.16) in section 4.2.4, except that the weights in this case are the product of the posterior probability $\gamma_{ir}^{(q)}$ of the group r and the posterior probability $\tau_{ijrk}^{(q)}$ of the polynomial component k for group r. The estimates are given by:

$$\begin{aligned}\boldsymbol{\beta}_{rk}^{(q+1)} &= \Big[\sum_{i=1}^{n}\sum_{j=1}^{m} \gamma_{ir}^{(q)} \tau_{ijrk}^{(q)} \mathbf{t}_j \mathbf{t}_j^T\Big]^{-1} \sum_{i=1}^{n}\sum_{j=1}^{m} \gamma_{ir}^{(q)} \tau_{ijrk}^{(q)} y_{ij} \mathbf{t}_j \\ &= (\mathbf{X}^{*T} \mathbf{W}_{rk}^{(q)} \mathbf{X}^*)^{-1} \mathbf{X}^{*T} \mathbf{W}_{rk}^{(q)} \mathbf{y}^*, \quad (4.54)\end{aligned}$$

where $\mathbf{W}_{rk}^{(q)}$ is the $nm \times nm$ diagonal matrix whose diagonal elements are the weights $\gamma_{ir}^{(q)} \tau_{ijrk}^{(q)}$ for all $i = 1, \ldots, n$ and $j = 1, \ldots, m$, that is:

$$\mathbf{W}_{rk}^{(q)} = \mathrm{diag}(\gamma_{1r}^{(q)} \tau_{11rk}^{(q)}, \ldots, \gamma_{1r}^{(q)} \tau_{1mrk}^{(q)}, \ldots, \gamma_{nr}^{(q)} \tau_{n1rk}^{(q)}, \ldots, \gamma_{nr}^{(q)} \tau_{nmrk}^{(q)}).$$

The maximization of $Q_{\boldsymbol{\theta}_{rk}}(\boldsymbol{\theta}_{rk}, \boldsymbol{\Psi}^{(q)})$ with respect to σ^2_{rk} also corresponds to K separate maximizations, or by equivalence minimizations where the criterion corresponding to component k $(k = 1, \ldots, K)$ is performed as

$$\sigma^{2(q+1)}_{rk} = \arg \min_{\sigma^2_{rk} \in \mathbb{R}^+} \sum_{i=1}^{n} \sum_{j=1}^{m} \gamma^{(q)}_{ir} \tau^{(q)}_{ijrk} \Big[\frac{(y_{ij} - \boldsymbol{\beta}^{T(q+1)}_{rk} \mathbf{t}_j)^2}{\sigma^2_{rk}} + \log \sigma^2_{rk} \Big] \tag{4.55}$$

which is also performed in a similar way as for (4.19) by adding the weights corresponding to the fuzzy group memberships that are the posterior group probabilities, which provides the following update formula by

$$\begin{aligned} \sigma^{2(q+1)}_{rk} &= \frac{1}{\sum_{i=1}^{n} \sum_{j=1}^{m} \gamma^{(q)}_{ir} \tau^{(q)}_{ijrk}} \sum_{i=1}^{n} \sum_{j=1}^{m} \gamma^{(q)}_{ir} \tau^{(q)}_{ijrk} (y_{ij} - \boldsymbol{\beta}^{T(q+1)}_{rk} \mathbf{t}_j)^2 \\ &= \frac{1}{\sum_{i=1}^{n} \sum_{j=1}^{m} \gamma^{(q)}_{ir} \tau^{(q)}_{ijrk}} (\mathbf{y}^* - \mathbf{X}^* \boldsymbol{\beta}^{(q+1)}_{rk})^T \mathbf{W}^{(q)}_{rk} (\mathbf{y}^* - \mathbf{X}^* \boldsymbol{\beta}^{(q+1)}_{rk}) \end{aligned} \tag{4.56}$$

Finally, the maximization of $Q_{\mathbf{w}_r}(\mathbf{w}_r, \boldsymbol{\Psi}^{(q)})$ with respect to $\mathbf{w}_r$ for each group r $(r = 1, \ldots, R)$ is a multinomial logistic regression problem weighted by $\gamma^{(q)}_{ir} \tau^{(q)}_{ijrk}$ which we solve with a multi-class IRLS algorithm.

The Iteratively Reweighted Least Squares (IRLS) algorithm The IRLS algorithm is used to maximize $Q_{\mathbf{w}_r}(\mathbf{w}_r, \boldsymbol{\Psi}^{(q)})$ with respect to the parameter $\mathbf{w}_r$ for each of the R groups in the M-step, at each iteration q of the EM algorithm. To estimate the parameter vector $\mathbf{w}_r = (\boldsymbol{w}_{r1}, \ldots, \boldsymbol{w}_{rK})$ the vector $\boldsymbol{w}_{rK}$ is set to the null vector to guarantee $\sum_{k=1}^{K} \pi_k(t_j; \mathbf{w}_r) = 1$. After starting with an initial solution $\mathbf{w}_r{}^{(0)}$, the estimation of $\mathbf{w}_r$ is updated iteratively where a single update at iteration l is given by

$$\mathbf{w}^{(l+1)}_r = \mathbf{w}^{(l)}_r - \Big[\frac{\partial^2 Q_{\mathbf{w}_r}(\mathbf{w}_r, \boldsymbol{\Psi}^{(q)})}{\partial \mathbf{w}_r \partial \mathbf{w}_r{}^T} \Big]^{-1}_{\mathbf{w}_r = \mathbf{w}^{(l)}_r} \frac{\partial Q_{\mathbf{w}_r}(\mathbf{w}_r, \boldsymbol{\Psi}^{(q)})}{\partial \mathbf{w}_r} \Big|_{\mathbf{w}_r = \mathbf{w}^{(l)}_r} . \tag{4.57}$$

The gradient vector of $Q_{\mathbf{w}_r}(\mathbf{w}_r, \boldsymbol{\Psi}^{(q)})$ consists of $K-1$ gradient component vectors where each component vector that corresponds to the kth component $\boldsymbol{w}_{rk}$ $(k = 1, \ldots, K-1)$ for $\mathbf{w}_r = \mathbf{w}^{(l)}_r$ is given by

$$\frac{\partial Q_{\mathbf{w}_r}(\mathbf{w}_r, \boldsymbol{\Psi}^{(q)})}{\partial \boldsymbol{w}_{rk}} \Big|_{\mathbf{w}_r = \mathbf{w}^{(l)}_r} = \sum_{i=1}^{n} \sum_{j=1}^{m} \gamma^{(q)}_{ir} \big(\tau^{(q)}_{ijrk} - \pi_k(t_j; \mathbf{w}^{(l)}_r) \big) \boldsymbol{v}_j \tag{4.58}$$

The Hessian matrix of $Q_{\mathbf{w}_r}(\mathbf{w}_r, \boldsymbol{\Psi}^{(q)})$ consists of $(K-1) \times (K-1)$ block matrices, each component evaluated at $\mathbf{w}_r = \mathbf{w}^{(l)}_r$ is given by

$$\frac{\partial^2 Q_{\mathbf{w}_r}(\mathbf{w}_r, \boldsymbol{\Psi}^{(q)})}{\partial \boldsymbol{w}_{rk} \partial \boldsymbol{w}^T_{r\ell}} \Big|_{\mathbf{w}_r = \mathbf{w}^{(l)}_r} = - \sum_{i=1}^{n} \sum_{j=1}^{m} \gamma^{(q)}_{ir} \pi_k(t_j; \mathbf{w}^{(l)}_r) \big(\delta_{k\ell} - \pi_\ell(t_j; \mathbf{w}^{(l)}_r) \big) \boldsymbol{v}_j \boldsymbol{v}^T_j , \tag{4.59}$$

for $\ell, k = 1, \ldots, K-1$. It can be see that the Gradient and the Hessian in this case are simply weighted variants of those given in Equation (4.25) and (4.23) respectively, where the weights are the fuzzy group memberships $\gamma^{(q)}_{ir}$.

The IRLS algorithm (4.57) provides the parameter $\mathbf{w}^{(q+1)}_r$ for each of the R groups. The pseudo code 5 summarizes the EM algorithm for the proposed MixRHLP model.

Algorithm 5 Pseudo code of the proposed algorithm for the MixRHLP model for a set of curves.

Inputs: training set of n curves $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ sampled at the time points $\mathbf{t} = (t_1, \ldots, t_m)$, the number of groups R, the number of polynomial components K for each of the R groups and the polynomial degree p

```
 1: Initialize: $\boldsymbol{\Psi}^{(0)} = (\alpha_1^{(0)}, \ldots, \alpha_R^{(0)}, \boldsymbol{\theta}_1^{(0)}, \ldots, \boldsymbol{\theta}_R^{(0)})$
 2: fix a threshold $\epsilon > 0$
 3: set $q \leftarrow 0$ (EM iteration)
 4: while increment in log-likelihood $> \epsilon$ do
 5:   E-Step:
 6:   for $r = 1, \ldots, R$ do
 7:     compute $\gamma_{ir}^{(q)}$ for $i = 1, \ldots, n$ using Equation (4.44)
 8:     for $k = 1, \ldots, K$ do
 9:       compute $\tau_{ijrk}^{(q)}$ for $i = 1, \ldots, n$ and $j = 1, \ldots, m$ using Equation (4.45)
10:     end for
11:   end for
12:   M-Step:
13:   for $r = 1, \ldots, R$ do
14:     compute the mixing proportion $\alpha_r^{(q+1)}$ using Equation (4.52)
15:     for for $k = 1, \ldots, K$ do
16:       compute $\boldsymbol{\beta}_{rk}^{(q+1)}$ using Equation (4.54)
17:       compute $\sigma_{rk}^{2(q+1)}$ using Equation (4.56)
18:     end for
19:     IRLS:
20:     Initialize: set $\mathbf{w}_r^{(l)} = \mathbf{w}_r^{(q)}$
21:     set a threshold $\delta > 0$
22:     $l \leftarrow 0$ (IRLS iteration)
23:     while increment in $Q_{\mathbf{w}_r}(\mathbf{w}_r, \boldsymbol{\Psi}^{(q)}) > \delta$ do
24:       compute $\mathbf{w}_r^{(l+1)}$ using Equation (4.57)
25:       $l \leftarrow l + 1$
26:     end while
27:     $\mathbf{w}_r^{(q+1)} \leftarrow \mathbf{w}_r^{(l)}$
28:     $q \leftarrow q + 1$
29:   end for
30: end while
31: $\hat{\boldsymbol{\Psi}} = (\alpha_1^{(q)}, \ldots, \alpha_R^{(q)}, \boldsymbol{\theta}_1^{(q)}, \ldots \boldsymbol{\theta}_R^{(q)})$
```

Output: $\hat{\boldsymbol{\Psi}}$ the maximum likelihood estimate of $\boldsymbol{\Psi}$

Model selection

The optimal number of groups or sub-classes R can be computed by maximizing the BIC criterion (Schwarz, 1978):

$$\mathrm{BIC}(R, K, p) = \mathcal{L}(\hat{\boldsymbol{\Psi}}) - \frac{\nu_R}{2}\log(n), \tag{4.60}$$

where $\hat{\boldsymbol{\Psi}}$ is the maximum likelihood estimate of the parameter vector $\boldsymbol{\Psi}$ provided by the EM algorithm, $\nu_R = R - 1 + \sum_{r=1}^{R}\nu_{\boldsymbol{\theta}_r}$ is the number of free parameters of MixRHLP model where $R-1$ represent the number of mixing proportions and $\nu_{\boldsymbol{\theta}_r} = (K(p+4)-2)$ represents the number of free parameters of the sub-model associated with group r, and n is the sample size.

4.3.4 Curves approximation with the MixRHLP model

With the proposed mixture modeling approach, each group (sub-class) of curves r is approximated by a single "mean" curve that is given by the conditional expectation $\mathbb{E}[\mathbf{y}_i|\mathbf{t}, h_i = r; \hat{\boldsymbol{\theta}}_r]$ as for the RHLP model presented in section 4.2. Each point of this curve is given by:

$$\begin{aligned}\hat{y}_{ij} &= \int_{\mathbb{R}} y_{ij} p(y_{ij}|h_i = r, t_j; \hat{\boldsymbol{\theta}}_r) dy_{ij}\\ &= \int_{\mathbb{R}} y_{ij} \sum_{k=1}^{K} \pi_k(t_j; \hat{\mathbf{w}}_r)\mathcal{N}\big(y_{ij}; \hat{\boldsymbol{\beta}}_{rk}^T\mathbf{t}_j, \hat{\sigma}_{rk}^2\big) dy_{ij}\\ &= \sum_{k=1}^{K} \pi_k(t_j; \hat{\mathbf{w}}_r)\hat{\boldsymbol{\beta}}_{rk}^T\mathbf{t}_j. \end{aligned}\tag{4.61}$$

The matrix formulation of the mean curve approximating the curves of the group r is then given by:

$$\hat{\mathbf{y}}_r = \sum_{k=1}^{K} \hat{\mathbf{\Pi}}_{rk}\mathbf{X}\hat{\boldsymbol{\beta}}_{rk} \tag{4.62}$$

where $\hat{\mathbf{\Pi}}_{rk} = \mathrm{diag}(\pi_k(t_1; \hat{\mathbf{w}}_r), \ldots, \pi_k(t_m; \hat{\mathbf{w}}_r))$ is a diagonal matrix whose diagonal elements are the logistic proportions associated with the kth regression model for the sub-class r.

The MixRHLP is naturally tailored to the unsupervised context for automatically searching for a homogeneous groups within a heterogeneous set of curves, that is the curve clustering problem.

4.3.5 Curve clustering with the MixRHLP model

In this section we use the MixRHLP model for clustering curves presenting regime changes. The use of the MixRHLP model in a curve clustering framework may arise in two different schemes, similarly as for the case of clustering multidimensional data with mixture models. The first one is referred to as the maximum likelihood estimation approach or the mixture approach. While the second approach is known as the classification likelihood approach or the classification approach. We derive the two approaches in the following sections.

The mixture approach

The principle of the estimation approach is to first estimate the model parameters in a maximum likelihood framework and then, given the estimated model parameters, find a partition of the curves into R clusters by applying the MAP rule on the estimated posterior probabilities. More precisely, given a set of unlabeled n curves which are assumed to be i.i.d generated according to the MixRHLP model presented in section 4.3, we first run the EM algorithm summarized by pseudo code 5 to obtain an estimation $\hat{\boldsymbol{\Psi}}$. Then, the cluster labels are determined from the posterior cluster probabilities (c.f., Equation (4.44)) obtained at convergence of the EM algorithm through the MAP decision rule, that is:

$$\hat{h}_i = \arg \max_{1 \leq r \leq R} \gamma_{ir}(\hat{\boldsymbol{\Psi}}) \quad (i = 1, \ldots, n) \tag{4.63}$$

where the notation $\gamma_{ir}(\hat{\boldsymbol{\Psi}})$ refers to the fact that the posterior probabilities γ_{ir} in (4.44) are computed with the parameter vector obtained at convergence of the EM algorithm 5.

The classification approach

The classification approach simultaneously performs the clustering and the parameter estimation by maximizing some classification log-likelihood criteria. The maximization is generally performed through the Classification EM (CEM) algorithm (Celeux and Govaert, 1992).

In the context of the MixRHLP model, we propose to maximize the classification log-likelihood criterion defined by:

$$\begin{aligned} CL(\boldsymbol{\Psi}, \mathbf{h}) &= \log p(\mathbf{Y}, \mathbf{h}|\mathbf{t}; \boldsymbol{\Psi}) = \log p(\mathbf{y}_1, \ldots, \mathbf{y}_n, h_1, \ldots, h_n|\mathbf{t}; \boldsymbol{\Psi}) \\ &= \log \prod_{i=1}^{n} \prod_{r=1}^{R} [p(h_i = r) p(\mathbf{y}_i | h_i = r, \mathbf{t}; \boldsymbol{\theta}_r)]^{h_{ir}} \\ &= \sum_{i=1}^{n} \sum_{r=1}^{R} h_{ir} \log [\alpha_r p(\mathbf{y}_i | h_i = r, \mathbf{t}; \boldsymbol{\theta}_r)] \\ &= \sum_{i=1}^{n} \sum_{r=1}^{R} h_{ir} \log \alpha_r + \sum_{i=1}^{n} \sum_{r=1}^{R} h_{ir} \log \prod_{j=1}^{m} \sum_{k=1}^{K} \pi_k(t_j; \mathbf{w}_r) \mathcal{N}\big(y_{ij}; \boldsymbol{\beta}_{rk}^T \mathbf{t}_j, \sigma_{rk}^2\big), \end{aligned} \tag{4.64}$$

with respect to the parameter vector $\boldsymbol{\Psi}$ and the curves partition $\mathbf{h}$. This criterion is the log-likelihood of the parameter vector $\boldsymbol{\Psi}$, given the observed data $\mathbf{Y}$ and the partition of the curves represented by the class labels $\mathbf{h}$.

In the following section, we describe the CEM algorithm (Celeux and Govaert, 1992) used to maximize the proposed classification criterion.

The CEM algorithm for the MixRHLP model

After starting with an initial model parameters $\mathbf{\Psi}^{(0)}$, each iteration of the CEM algorithm consists of computing a partition of the curves into R clusters and updating the model parameters from the obtained partition. More specifically, in this case, the CEM algorithm starts with an initial parameter vector $\mathbf{\Psi}^{(0)}$ and iteratively alternates between the following steps until convergence:

Step 1: Compute the missing cluster labels $\mathbf{h}^{(q+1)}$ for the current estimated model parameters $\mathbf{\Psi}^{(q)}$ using the MAP decision rule:

$$\mathbf{h}^{(q+1)} = \arg \max_{\mathbf{h} \in \{1,\ldots,R\}^n} CL(\mathbf{\Psi}^{(q)}, \mathbf{h}), \tag{4.65}$$

which is equivalent to

$$h_i^{(q+1)} = \arg \max_{1 \leq r \leq R} \gamma_{ir}^{(q)} \ (i = 1, \ldots, n) \tag{4.66}$$

where $\gamma_{ir}^{(q)}$ is the posterior probability that the ith curve belongs to cluster r (c.f., Equation (4.44).

Step 2: Compute the model parameters update $\mathbf{\Psi}^{(q+1)}$, given the current cluster labels $\mathbf{h}^{(q+1)}$. This is achieved by maximizing the classification log-likelihood function w.r.t $\mathbf{\Psi}$:

$$\mathbf{\Psi}^{(q+1)} = \arg \max_{\mathbf{\Psi}} CL(\mathbf{\Psi}, \mathbf{h}^{(q+1)}). \tag{4.67}$$

As it can be seen in (4.64), the maximization of $CL(\mathbf{\Psi}, \mathbf{h}^{(q+1)})$ w.r.t $\mathbf{\Psi}$ can be performed by separately maximizing

$$CL_\alpha(\alpha_1, \ldots, \alpha_R, \mathbf{h}^{(q+1)}) = \sum_{i=1}^{n} \sum_{r=1}^{R} h_{ir}^{(q+1)} \log \alpha_r$$

w.r.t $(\alpha_1, \ldots, \alpha_R)$, and

$$CL_{\boldsymbol{\theta}}(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_R, \mathbf{h}^{(q+1)}) = \sum_{r=1}^{R} \sum_{i=1}^{n} h_{ir}^{(q+1)} \sum_{j=1}^{m} \log \sum_{k=1}^{K} \pi_k(t_j; \mathbf{w}_r) \mathcal{N}\big(y_{ij}; \boldsymbol{\beta}_{rk}^T \mathbf{t}_j, \sigma_{rk}^2\big)$$

w.r.t $(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_R)$ where $\boldsymbol{\theta}_r = (\mathbf{w}_r, \boldsymbol{\beta}_{r1}, \ldots, \boldsymbol{\beta}_{rK}, \sigma_{r1}^2, \ldots, \sigma_{rK}^2)$ is the parameter vector of the rth RHLP model.

Hence, the maximization of CL_α with respect to the mixing proportions $(\alpha_1, \ldots, \alpha_R)$ is performed directly in a similar way as for the maximization (4.49) in the case of the EM algorithm for the MixRHLP model (c.f., paragraph 4.3.3 in section 4.3.3). The updating formula is therefore given by:

$$\alpha_r^{(q+1)} = \frac{1}{n} \sum_{i=1}^{n} h_{ir}^{(q+1)} \quad (r = 1, \ldots, R). \tag{4.68}$$

The maximization of $CL_{\boldsymbol{\theta}}$ consists itself of R separate maximization problems, each of them consisting of maximizing

$$CL_{\boldsymbol{\theta}_r}(\boldsymbol{\theta}_r, \mathbf{h}^{(q+1)}) = \sum_{i|h_i^{(q+1)}=r}^{n} \sum_{j=1}^{m} \log \sum_{k=1}^{K} \pi_k(t_j; \mathbf{w}_r) \mathcal{N}\big(y_{ij}; \boldsymbol{\beta}_{rk}^T \mathbf{t}_j, \sigma_{rk}^2\big)$$

with respect to $\boldsymbol{\theta}_r$. This maximization can not be performed in a closed form. The difficulty arises from the presence of the summation over k that appears inside the logarithm, which is due to the fact that process $\mathbf{z}_r$ that control the regime changes within each cluster of curves is still missing. However, it can be seen that the criterion $CL_{\boldsymbol{\theta}_r}$ is none other than the log-likelihood function for the RHLP model given the set of curves of cluster r (c.f., Equation 4.8). Therefore, this problem can be solved by applying the EM algorithm for the RHLP model detailed in section 4.2.4. The corresponding parameter update $\boldsymbol{\theta}_r^{(q+1)}$ is then taken at convergence of the EM algorithm implemented by pseudo code 4, the algorithm being initialized with $\boldsymbol{\theta}_r^{(q)}$ obtained at the previous CEM iteration.

By construction, this CEM algorithm always monotonically increases the classification log-likelihood $CL(\boldsymbol{\Psi}, \mathbf{h})$ (c.f., Equation (4.64)) and convergence is guaranteed toward a local maximum of the optimized function in a finite number of iterations, since there are only a finite number of curves partitions (Celeux and Govaert, 1992). Notice that the number of EM iterations in the second step of the CEM algorithm can be limited. In this case, the obtained algorithm can be seen as a “generalized CEM” algorithm which has the same properties as the CEM algorithm described above.

Algorithm 6 gives the pseudo code of the CEM algorithm for curve clustering by using the proposed MixRHLP model.

Algorithm 6 Pseudo code of the CEM algorithm for the proposed MixRHLP model.

Inputs: training set of n curves $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$, the number of clusters R, the number of polynomial components K, the polynomial degree p and the sampling time $\mathbf{t} = (t_1, \ldots, t_m)$

```
1:  Initialize: $\boldsymbol{\Psi}^{(0)} = (\alpha_1^{(0)}, \ldots, \alpha_R^{(0)}, \boldsymbol{\theta}_1^{(0)}, \ldots, \boldsymbol{\theta}_R^{(0)})$
2:  while increment in log-likelihood $> \epsilon$ do
3:    Step 1:
4:    for $r = 1, \ldots, R$ do
5:      compute $\gamma_{ir}^{(q)}$ for $i = 1, \ldots, n$ using Equation (4.44)
6:    end for
7:    for $r = 1, \ldots, R$ do
8:      compute the cluster labels $h_i^{(q+1)}$ for $i = 1, \ldots, n$ using (4.66)
9:    end for
10:   Step 2:
11:   for $r = 1, \ldots, R$ do
12:     compute $\boldsymbol{\theta}_r^{(q+1)}$ using the procedure RHLP($\{\mathbf{y}_i | h_i^{(q+1)} = r\}$,$\mathbf{t}$, $K$,$\boldsymbol{\theta}^{(q)}$) implemented by Algorithm 4, Algorithm 4 being initialized with $\boldsymbol{\theta}^{(q)}$.
13:   end for
14:   $q \leftarrow q + 1$
15: end while
16: $\hat{\boldsymbol{\Psi}} = \boldsymbol{\Psi}^{(q)}$
17: $\hat{\mathbf{h}} = \mathbf{h}^{(q)}$
```

Outputs: $\hat{\boldsymbol{\Psi}} = (\hat{\alpha}_1, \ldots, \hat{\alpha}_R, \hat{\boldsymbol{\theta}}_1, \ldots, \hat{\boldsymbol{\theta}}_R)$; $\hat{\mathbf{h}} = (\hat{h}_1, \ldots, \hat{h}_n)$

The MixRHLP model for curve clustering can naturally be used in a supervised context, to perform curve classification, in particular in the case of complex shaped classes of curves. In the next section we will develop the classification rule resulting from the MixRHLP model.

4.3.6 Curve classification with the MixRHLP model

This section describes the use of the MixRHLP model in curve classification. In particular, we consider classes which may have complex shapes. In that heterogeneous case, the FLDA classification rule (c.f., section 4.2.6), which is based on a single RHLP model for each class of curves, may has limited performance. The mixture formulation of the RHLP model can therefore handle this problem.

Modeling the classes of curves with the MixRHLP model

Suppose we have a labeled training set of n curves $(\mathbf{Y}, \mathbf{c}) = ((\mathbf{y}_1, c_1), \ldots, (\mathbf{y}_n, c_n))$ where the class label c_i of the ith curve is in $\{1, \ldots, G\}$, G being the number of classes. In this section we assume that each class of curves g $(g = 1, \ldots, G)$ has a complex shape so that it is composed of R_g homogeneous sub-classes. Each sub-class r ($r = 1, \dots, R_g$) of class g is governed by K_g polynomial regimes. The mixture of regression models with hidden logistic processes (MixRHLP), is used to accurately model each class g. Therefore, the distribution of a curve $\mathbf{y}_i$ issued from class g is defined, as in Equation (4.35), by:

$$p(\mathbf{y}_i|c_i = g, \mathbf{t}; \mathbf{\Psi}_g) = \sum_{r=1}^{R_g} \alpha_{gr} \prod_{j=1}^{m} \sum_{k=1}^{K_g} \pi_k(t_j; \mathbf{w}_{gr}) \mathcal{N}(y_{ij}; \boldsymbol{\beta}_{grk}^T \mathbf{t}_j, \sigma_{grk}^2). \tag{4.69}$$

where $\alpha_{gr} = p(h_i = r|c_i = g)$ is the prior probability of the sub-cluster r of class g and $\mathbf{\Psi}_g = (\alpha_{g1}, \dots, \alpha_{gR_g}, \boldsymbol{\theta}_{g1}, \dots, \boldsymbol{\theta}_{gR_g})$ is the parameter vector of class g, $\boldsymbol{\theta}_{gr} = (\mathbf{w}_{gr}, \boldsymbol{\beta}_{gr1}, \dots, \boldsymbol{\beta}_{grK}, \sigma_{gr1}^2, \dots, \sigma_{grK}^2)$ being the parameter vector of each sub-class r of class g.

The classification rule derived from this approach can be seen as a Functional Mixture Discriminant Analysis (FMDA) classification rule particularly adapted to curves with changes in regimes, as in Gui and Li (2003) (c.f., section 2.8.2). The approach described here takes into account both the class dispersion, as it uses a mixture of specific regression models to represent each complex shaped class of curves, and the regimes involved in each sub-class of curves through the RHLP model used for each sub-class. The classification rule based on the MixRHLP model is given in the next section.

Curve classification rule

Based on the mixture modeling approach described above, the following MAP classification rule for curves is derived. First we learn the classes' parameters by applying the EM algorithm implemented by pseudo code 5 to each class of curves. This learning step provides the maximum likelihood estimates $(\hat{\mathbf{\Psi}}_1, \dots, \hat{\mathbf{\Psi}}_G)$ of the classes' parameters. A new curve $\mathbf{y}_i$ is then assigned to the class $\hat{c}_i$ maximizing the posterior probabilities:

$$\hat{c}_i = \arg \max_{1 \leq g \leq G} p(c_i = g|\mathbf{y}_i, \mathbf{t}; \hat{\mathbf{\Psi}}_g), \quad (i = 1, \dots, n) \tag{4.70}$$

where

$$p(c_i = g|\mathbf{y}_i, \mathbf{t}; \hat{\mathbf{\Psi}}_g) = \frac{w_g p(\mathbf{y}_i|c_i = g, \mathbf{t}; \hat{\mathbf{\Psi}}_g)}{\sum_{g'=1}^{G} w_{g'} p(\mathbf{y}_i|c_i = g', \mathbf{t}; \hat{\mathbf{\Psi}}_{g'})} \tag{4.71}$$

is the posterior probability of class g ($g = 1, \dots, G$) and $w_g = p(c_i = g)$ is its probability, estimated by the proportion of the curves belonging to class g in the training set.

The optimal number of sub-classes R_g for each class g ($g = 1, \dots, G$) can be computed by maximizing the BIC criterion defined in section 4.3.3.

Remark: Link with the polynomial regression mixture model

The polynomial regression mixture model (Gaffney, 2004) arises when the MixRHLP model is defined with a single regime ($K = 1$) for each cluster of curves.

4.4 Experiments on simulated curves

In this section, we study the performance of the methods developed in Chapters 3 and 4, which are the RHLP model for a single curve and for a set of curves, and the MixRHLP model for an heterogeneous set of curves. The models are evaluated in terms of curve modeling, segmentation, classification and clustering, in a series of experiments conducted on synthetic curves.

Evaluation criteria

Three evaluation criteria are used in the simulations to judge performance of the RHLP model for a single curve as well as for a set of curves.

The first criterion is the mean square error between the true simulated curve without noise (the true "mean" curve) and the estimated curve. This criterion will be referred to as *approximation error* and is used to assess the approaches with regard to curve modeling. It is computed by the formula

$$\frac{1}{m}\sum_{j=1}^{m}(\mathbb{E}[y_{ij}|t_j;\boldsymbol{\theta}] - \hat{y}_j)^2$$

where $\boldsymbol{\theta}$ is the true parameter. Let us recall that each point $\hat{y}_j$ of the estimated curve is given by:

- $\hat{y}_j = \sum_{k=1}^{K} \pi_k(t_j;\hat{\mathbf{w}})\hat{\boldsymbol{\beta}}_k^T\mathbf{t}_j$ for the proposed regression model with hidden logistic process (RHLP) model;
- $\hat{y}_j = \sum_{k=1}^{K} \hat{z}_{jk}\hat{\boldsymbol{\beta}}_k^T\mathbf{t}_j$ for the piecewise polynomial regression (PWPR) model;
- $\hat{y}_j = \sum_{k=1}^{K} \tau_{jk}(\hat{\boldsymbol{\Psi}})\hat{\boldsymbol{\beta}}_k^T\mathbf{t}_j$ for the Hidden Markov Model Regression (HMMR).

The second criterion is the segmentation error rate between the simulated and the estimated segmentation of a curve into K polynomial regimes. It is used to assess the models with regard to their performance to accurately segment the curves according to the true underlying regimes.

The third criterion is the curve misclassification error rate computed by a 5-fold cross-validation procedure. The curve classification based on the RHLP model for a single curve is performed by MDA in the space of descriptors. When the RHLP or the MixRHLP is used for a set of curves, the curve classification is performed directly in the curve space.

In addition, the average running time is given for each approach.

Initialization strategies and stopping rules

The proposed EM algorithm for the RHLP model and the EM (Baum-Welch) algorithm for Hidden Markov Model Regression are initialized as follows:

- To initialize $\boldsymbol{\beta}_k$ and σ_k^2 for $k = 1, \ldots, K$, a random segmentation of the curve into K segments is used. Then, we fit a polynomial to each segment k and then we deduce the value of σ_k^2. We use 10 runs of EM and the solution providing the highest likelihood is chosen.
- The internal IRLS algorithm is initialized randomly with a random vector $\mathbf{w}$ only for the first EM iteration ($q = 0$). From the second EM iteration, it is initialized with $\mathbf{w}^{(q)}$ obtained at the previous EM iteration.
- In the HMMR the initial probabilities are set to $\pi = (1, 0, \ldots, 0)$ and the initial transition probabilities are set to $\mathbf{A}_{\ell k} = 0.5$ for $\ell \leq k \leq \ell + 1$;

All the EM algorithms are stopped when the relative variation of the optimized log-likelihood function between two iterations is below a predefined threshold, that is $|\frac{\mathcal{L}^{(q+1)} - \mathcal{L}^{(q)}}{\mathcal{L}^{(q)}}| \leq 10^{-6}$ or when the iteration number reaches 1000. The internal IRLS loop is stopped when the relative variation of $Q_{\mathbf{w}}$, that is $|\frac{Q_{\mathbf{w}}(\mathbf{w}^{(l+1,q)}) - Q_{\mathbf{w}}(\mathbf{w}^{(l,q)})}{Q_{\mathbf{w}}(\mathbf{w}^{(l,q)})}|$, is below 10^{-6} or when the iteration number reaches 50.

For the HMMR model used here, we assume that the underlying regimes are ordered in the time. In order to obtain contiguous segments, we impose the following constraints: $p(z_j = k | z_{j-1} = \ell) = 0 \quad \text{if} \quad k < \ell$ and $p(z_j = k | z_{j-1} = \ell) = 0 \quad \text{if} \quad k > \ell + 1$. This corresponds to a particular model called left-right HMM (see section 2.5.3).

4.4.1 Evaluation in terms of curve modeling

In this section we evaluate, in terms of curve modeling and segmentation, the proposed RHLP model for a single curve and for a set of curves by comparing it to the following alternative models:

- the polynomial spline regression (PSR) model (c.f., section 2.6.2 and section 2.6.3);
- the piecewise polynomial regression (PWPR) model (c.f., section 2.6.4 and section 2.6.5);
- the Hidden Markov Model Regression (HMMR) (c.f., section 2.6.6). We note that, here, the HMMR model only concerns the case of a single curve.

Effect of the smoothness level of transition

The experiment performed in this section aims at observing the effect of the smoothness level of transitions on estimation quality.

In this case, each curve is composed of $m = 100$ points and is generated according to the RHLP model with three constant polynomial regimes ($K = 3, p = 0$) and regime transitions at 1 and 3 seconds. The used simulation parameters for this situation are shown in Table 4.1.

$\boldsymbol{\beta}_1 = 0$	$\boldsymbol{w}_1 = [3341.33, -1706.96]$	$\sigma_1 = 2$
$\boldsymbol{\beta}_2 = 10$	$\boldsymbol{w}_2 = [2436.97, -810.07]$	$\sigma_2 = 2$
$\boldsymbol{\beta}_3 = 5$	$\boldsymbol{w}_3 = [0, 0]$	$\sigma_3 = 2$

Table 4.1: Simulation parameters for experiment 1 corresponding to varying the smoothness level of transitions.

The smoothness level of transitions is tuned by the parameter $\lambda_k = \boldsymbol{w}_{k1}$ seen in section 3.2.1 and Figure 3.3 (a). We consider 10 decreasing values of $|\lambda_k|$, which correspond to 10 increasing values of the smoothness level of regime changes (see Table 4.2). Figure 4.5 (a) shows the true curves without noise for the 10 smoothness levels

Smoothness level of transitions	1	2	3	4	5	6	7	8	9	10
$\lvert\lambda_k\rvert$	$\frac{\lvert\boldsymbol{w}_{k1}\rvert}{1}$	$\frac{\lvert\boldsymbol{w}_{k1}\rvert}{2}$	$\frac{\lvert\boldsymbol{w}_{k1}\rvert}{5}$	$\frac{\lvert\boldsymbol{w}_{k1}\rvert}{10}$	$\frac{\lvert\boldsymbol{w}_{k1}\rvert}{20}$	$\frac{\lvert\boldsymbol{w}_{k1}\rvert}{40}$	$\frac{\lvert\boldsymbol{w}_{k1}\rvert}{50}$	$\frac{\lvert\boldsymbol{w}_{k1}\rvert}{80}$	$\frac{\lvert\boldsymbol{w}_{k1}\rvert}{100}$	$\frac{\lvert\boldsymbol{w}_{k1}\rvert}{125}$

Table 4.2: The different smoothness levels from abrupt transitions to smooth transitions for the situations shown in Figure 4.5.

of transitions and Figure 4.5 (b) shows an example of simulated curves for a fixed smoothness level.

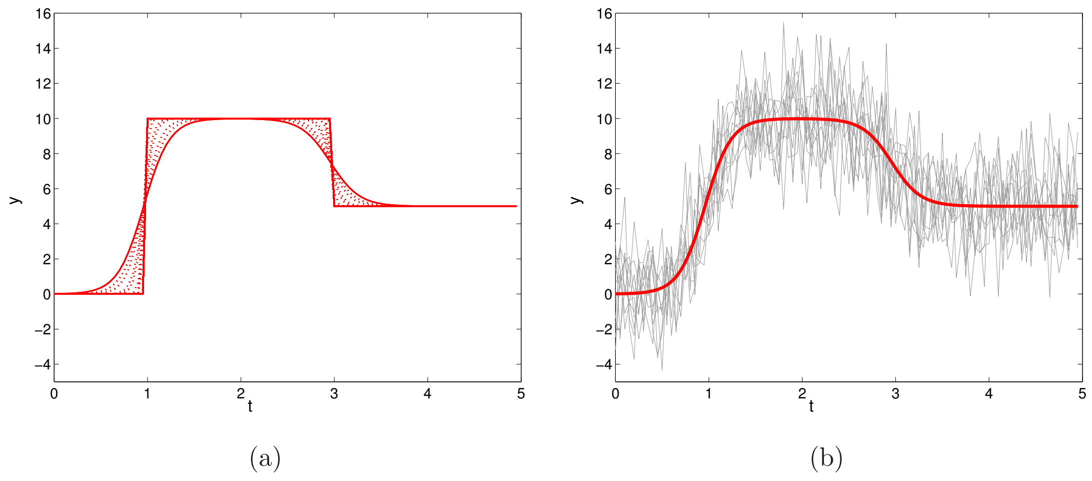


Figure 4.5: Example of 50 curves where the mean curve is simulated according to the proposed regression model with a curve size $m = 100$.

For each value of the smoothness level of transitions, the assessment criteria are averaged over 20 different data sets. For the spline models, we use continuous piecewise linear functions (linear splines) with 8 internal knots placed uniformly. The choice of a linear (B) spline rather than other spline models, is due the fact that the considered regimes are constant.

Obtained results

Figure 4.6 shows the curve approximation error in relation to the smoothness level of transitions, obtained with the four models. It can be seen that the proposed approach provides more accurate curve modeling results compared to the alternative approaches. In particular, one can clearly observe that the spline regression models are more adapted for approximating smooth curves rather than for curves with abrupt regime changes. One can observe that, the approximation error provided by the spline models is decreasing as the curves tend to be smooth, in particular for the three last smoothness levels of transitions.

On the other hand, the PWPR and the HMMR approaches provide closely similar results to those provided by the proposed RHLP model when the regime changes are abrupt (levels 1, 2 and 3). However, compared to the alternative approaches, the RHLP model provides significantly better results when the regime changes become smooth.

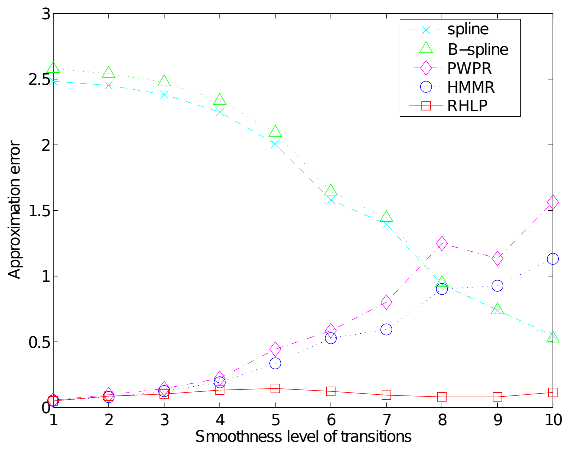


Figure 4.6: Approximation error in relation to the smoothness level of transitions for situation 1

Effect of the sample size n, the curve size m and the noise level σ

In this section, we conduct three series of experiments on simulated curves in order to observe the effect of the sample size n, the curve size m and the noise level σ, on estimation quality.

The first series of experiments consists of varying the sample size n from 10 to 100 by step of 10, the curve size being fixed to $m = 100$. For the second experiment, to observe the effect of the curve size m on estimation quality, we vary the curve size m ($m = 100, 300, 500, 700, 1000, 1500$), the sample size being fixed to $n = 10$. The third experiment aims at observing the effect of the noise level σ on modeling accuracy. In this experiment, the noise level σ is assumed to be common for all the segments and varies from 0.5 to 5 by step of 0.5, the curve size in this case being set to $m = 500$. We consider that all the curves are regularly sampled over 5 seconds. For each of the three series of experiments, three situations are considered to evaluate the proposed approaches.

- In the first situation, the curves are simulated with the proposed RHLP model. The simulated curves consisted of K polynomial regimes ($K = 4$) with a polynomial degree $p = 2$ and transitions at $t \in \{1, 3, 4\}$ seconds.
- The second situation consists of a two-component polynomial piecewise function with a polynomial degree $p = 2$ and a transition at $t = 2.5$ second.
- For the third situation, each curve is generated from an arbitrary non-linear function corrupted by a noise.

Table 4.3 shows the set of simulation parameters used for the three situations and Figure 4.7 shows an example of simulated curve for each of the three situations.

True curve	Parameters	
Situation 1: $\sum_{k=1}^{4} \pi_k(t;\mathbf{w})\boldsymbol{\beta}_k^T\mathbf{t}$	$\boldsymbol{\beta}_1 = [34, -60, 30]$	$\boldsymbol{w}_1 = [547, -154]$
	$\boldsymbol{\beta}_2 = [-17, 29, -7]$	$\boldsymbol{w}_2 = [526, -135]$
	$\boldsymbol{\beta}_3 = [185, -104, 15]$	$\boldsymbol{w}_3 = [464, -115]$
	$\boldsymbol{\beta}_4 = [-804, 343, -35]$	$\boldsymbol{w}_4 = [0, 0]$
Situation 2: $\boldsymbol{\beta}_1^T\mathbf{t}\mathbb{1}_{[0;2.5]}(t) + \boldsymbol{\beta}_2^T\mathbf{t}\mathbb{1}_{]2.5;5]}(t)$	$\boldsymbol{\beta}_2 = [-78, 47, -5]$	
	$\boldsymbol{\beta}_1 = [33, -20, 4]$	
Situation 3: $20\sin(1.6\pi t)\exp(-0.7t)$		

Table 4.3: Simulation parameters for the three situations

For the used spline models, the knots are placed uniformly in the range of t. We use a quadratic spline model for situation 2 with 3 internal knots as the two polynomial regimes are of degree 2. For the two other situations, we use cubic spline models with 9 internal knots. The number of knots for the two situations is chosen manually in accordance to the nonlinearity of the function to be estimated.

Obtained results

In this section we report the modeling results obtained by the four approaches. For each series of experiments, the approximation error represent an average over 20 different data sets.

The curve modeling error in relation to the number of curves n, the curve size m and the noise level σ, is shown in Figure 4.8. It can be seen that the errors decrease when the curve size and the sample size increase. The modeling results provided by the proposed RHLP model are more accurate compared to those obtained by the alternative approaches. For the results corresponding to the second situation (see Figure 4.8 (middle)), the piecewise model is competitive with the RHLP model. This is attributed to the fact that the function in this case is simulated according to a piecewise continuous function and therefore the two approaches tend to have closely similar performance.

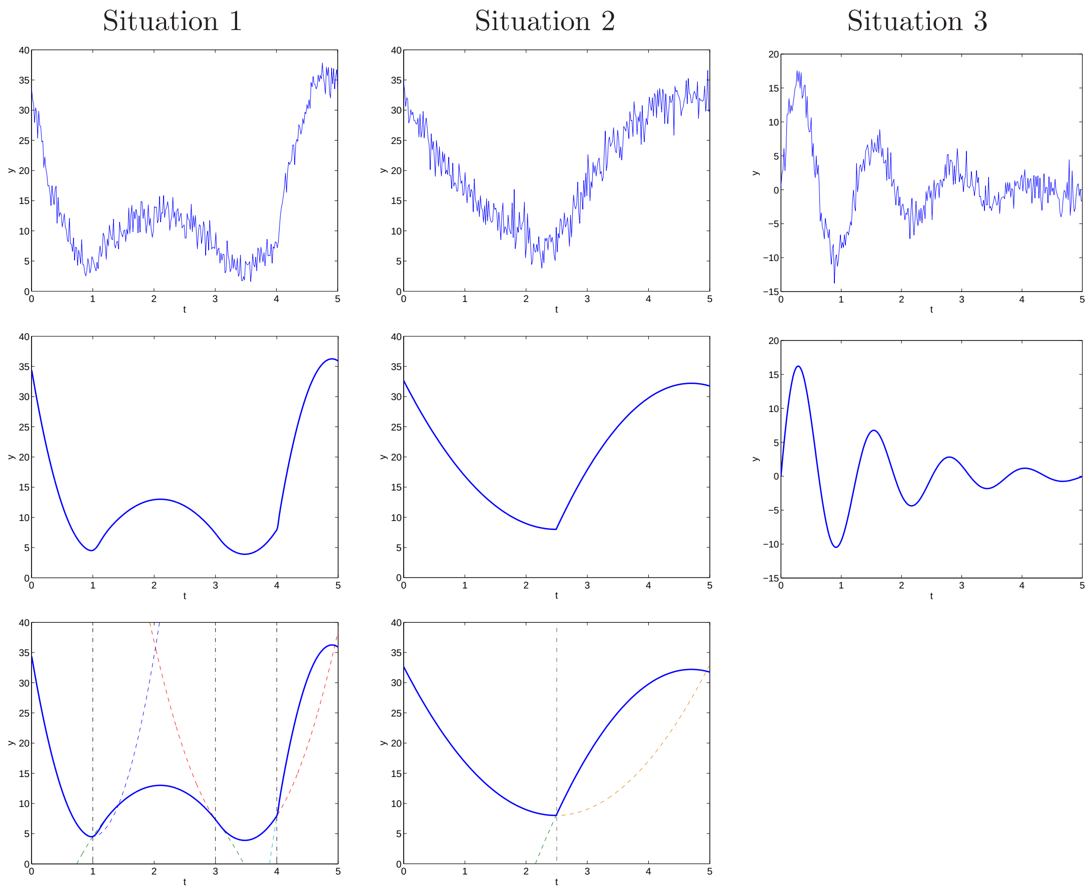


Figure 4.7: Example of simulated curves for a curve size $m = 300$ with the noisy curves (top) and the true mean curves (middle) fro each situation. The mean curves and the corresponding polynomial components, and the transition points are shown in the bottom plots. The curve in the left plot is simulated curve according to the RHLP model. The middle plots show the piecewise polynomial noisy curve simulated according to the PWPR model and the right plot shows the nonlinear function (c.f., Table 4.3).

On the other hand, when the noise level increases, the proposed RHLP model is more stable than the considered regression models, as it can seen in Figure 4.8 (right).

4.4.2 Evaluation in terms of running time

In this section we highlight the practical efficiency of the algorithm for the proposed approach in terms of running time comparing the alternative algorithms.

The running time results reported here represent averages over 20 different data sets for each situation, for each value of n and for each value of m. For each data set, the CPU running time of the EM algorithm for the proposed RHLP model represents

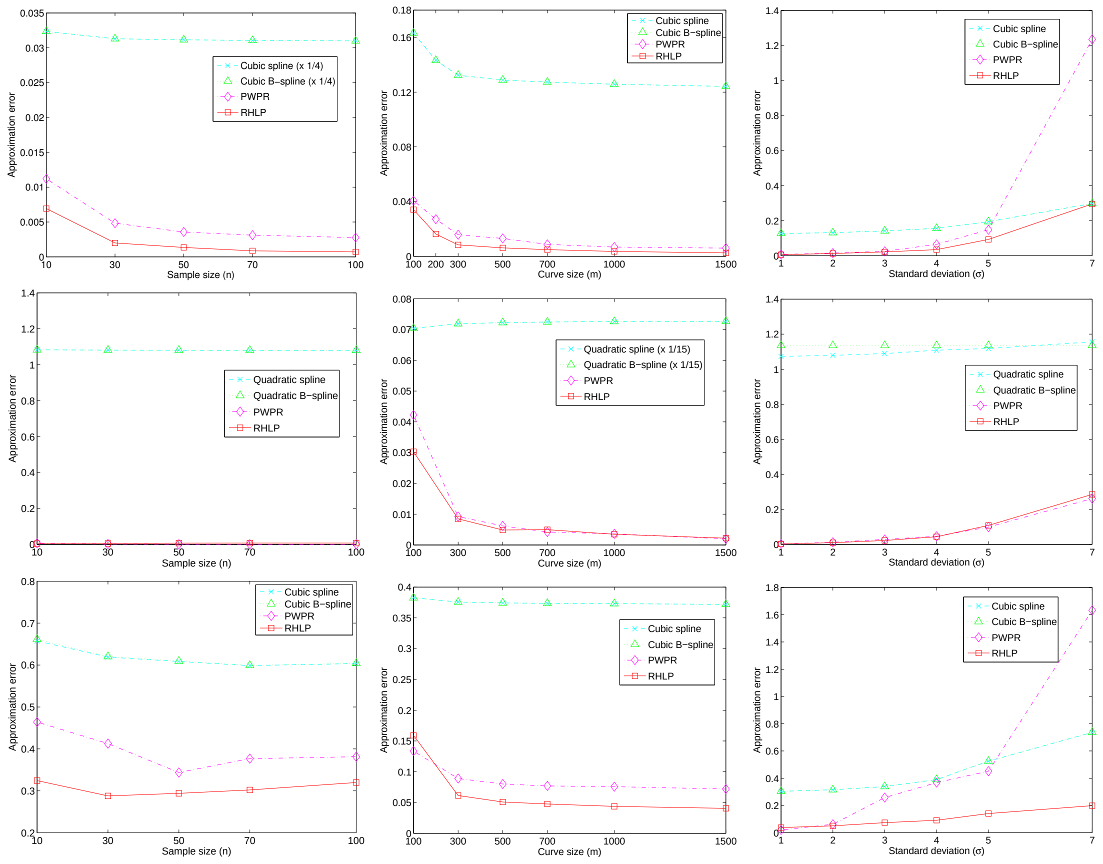


Figure 4.8: Approximation error rate in relation to the number of curves n for $m = 500$ and $\sigma = 1.5$ (left) , in relation to the curve size m for $n = 10$ curves and $\sigma = 1.5$ (middle), and in relation to the standard deviation σ for $n = 10$ and $m = 500$ (right). From top to bottom, the plots show the results corresponding to situation 1, situation 2 and situation 3.

are averaged over 10 runs of the EM algorithm.

The compared models are the proposed RHLP model and the piecewise polynomial regression model. The spline regression models are not considered in the comparisons as they are estimated in a closed form and their computational time are close to zero. The HMMR model which is not adapted for a set of curves, was only used for the case of a single curve. The CPU running times of the used EM (Baum-Welch) algorithm for the HMMR model are quasi identical to those of the EM algorithm for the RHLP model, but are not reported here.

Figure 4.9 shows the average running time in seconds for the PWPR approach, which uses dynamic programming, and for the proposed approach which uses the EM algorithm. One can clearly observe that the piecewise approach tends to need an expensive computational load as the curve size and the number of curves increase. It can be also seen that the EM algorithm RHLP model is significantly faster than the dynamic programming procedure used for piecewise regression.

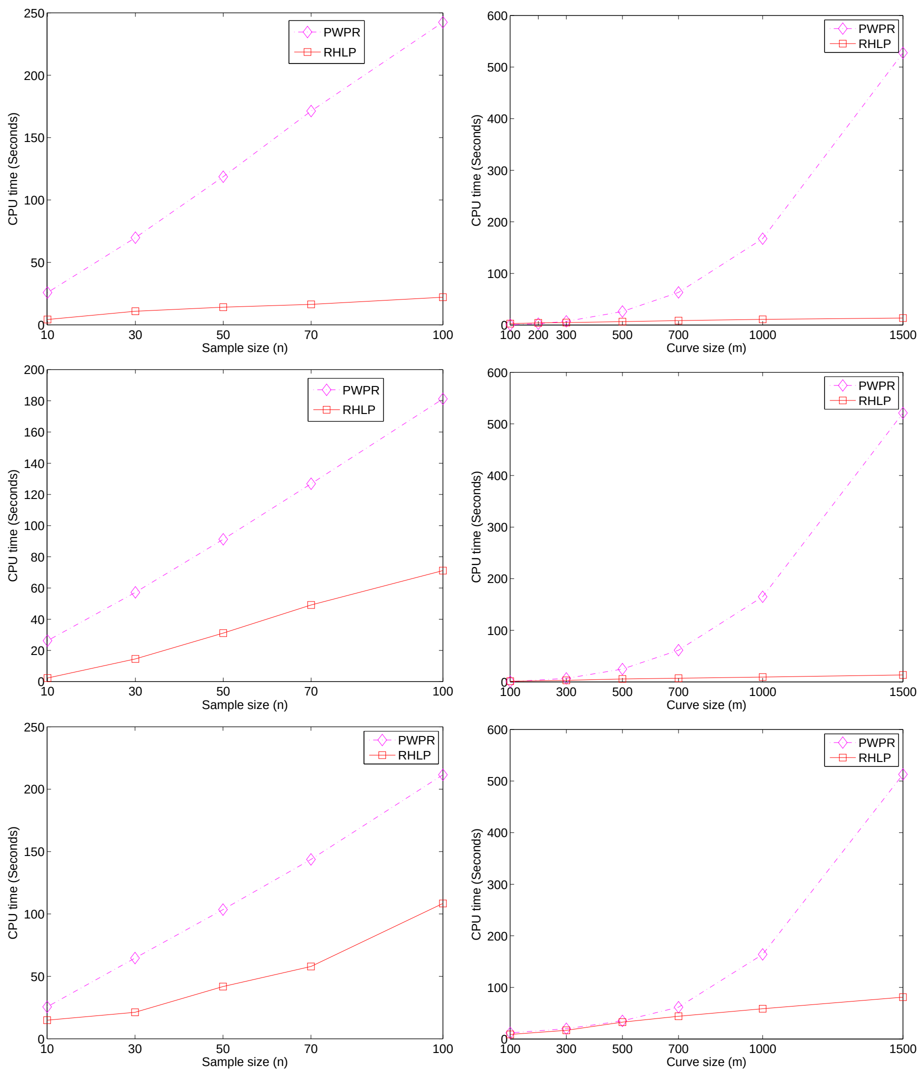


Figure 4.9: Average CPU time in relation to the curve size n for $m = 500$ (left) and in relation to the curve size m for $n = 10$ (right) obtained for situation 1 (top), situation 2 (middle) and situation 3 (bottom).

4.4.3 Evaluation in terms of curve segmentation

Simulation protocol

In addition to the two situations (situation 1 and 2) considered in the previous section (see Table 4.3), we consider another situation corresponding to a three-component piecewise constant function with abrupt regime changes. Figure 4.10 shows an example of a simulated curve corresponding to this situation.

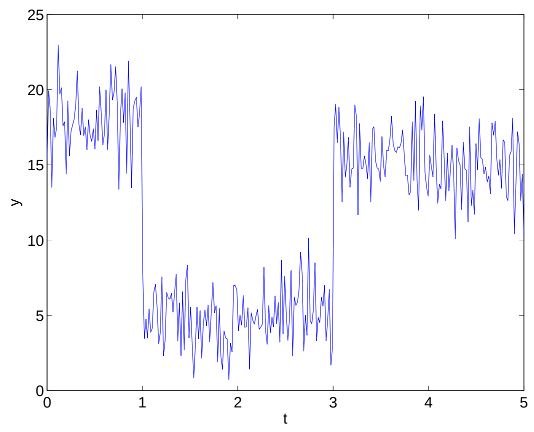


Figure 4.10: Situation 4: example of a simulated curve according a piecewise constant function with $y_t = 18 \times \mathbb{1}_{[0;1]}(t) + 5 \times \mathbb{1}_{]1;3]}(t) + 15 \times \mathbb{1}_{]3;5]}(t) + \epsilon_t$, $\epsilon_t \sim \mathcal{N}(0, 2^2)$.

Obtained results

Here we report the curve segmentation results obtained by the proposed approach (RHLP), the Markovian approach (HMMR) and the piecewise polynomial approach (PWPR) for situation 1, 2 and 4. The spline approaches which are not dedicated to curve segmentation are not considered for comparisons.

Figure 4.11 shows the curve segmentation results in relation to the curve size m and the noise level σ obtained by the three approaches. It can be seen that the segmentation error decreases when the curve size m increases, except for the last situation for which the segmentation error is constant, since the transitions are very pronounced (c.f. Figure 4.10).

For the two first situations, the proposed approach provides more accurate segmentation results than the two alternative approaches, as it can be observed in the top and the middle plots of Figure 4.11. In addition, when the noise level increases, the RHLP model remains more stable than to the PWPR and the HMMR models as it provides the lower segmentation error (see the top right and the middle right plots of Figure 4.10).

On the other hand, the results are quasi identical for the piecewise constant curves with abrupt regime changes corresponding to the last situation (see Figure 4.11 (bottom). For this situation, one can observe that the three approaches have quasi identical performance in curve segmentation as the regimes are very pronounced. In addition, when the curve size increases, the segmentation error remains close to zero (see the bottom-left plot in Figure 4.11). The performance of the piecewise regression model can be better, compared to the two other approaches, for a high noise level (see the bottom-right plot of Figure 4.11).

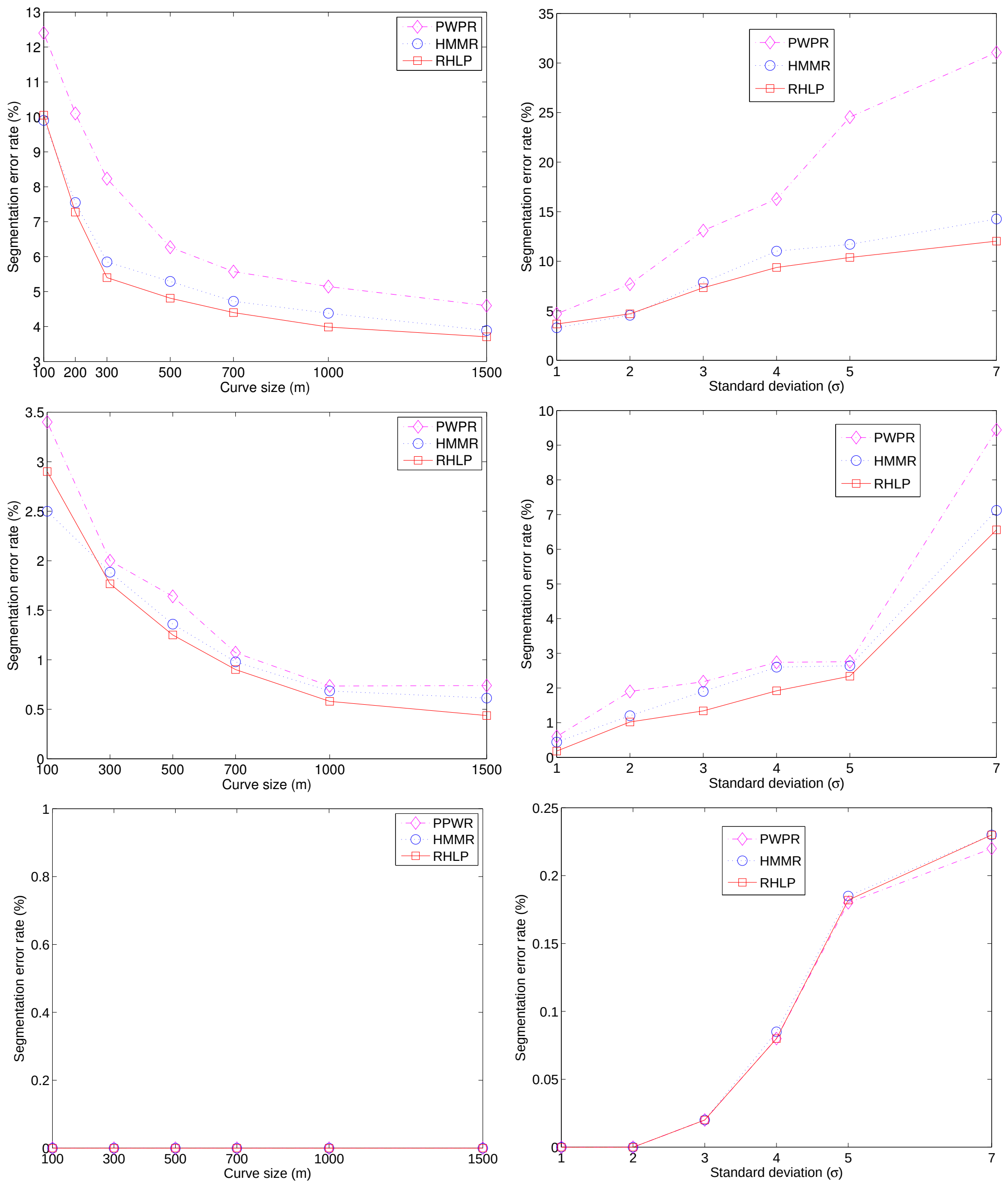


Figure 4.11: Segmentation error rate in relation to the curve size m (left) for $n = 10$ curves and $\sigma = 1.5$, and in relation to the standard deviation σ for $n = 10$ and $m = 500$ (right) obtained for situation 2 (top) and situation 4 (bottom).

4.4.4 Evaluation in terms of curve classification

This part is devoted to the evaluation of the classification accuracy of the proposed approach based on the RHLP model by using simulated curves. Comparisons are performed with the alternative classification approaches based on polynomial regression (PR), polynomial spline regression (PSR) and piecewise polynomial regression (PWPR). Each approach consists of a model estimation followed by MAP in the space of curves. The classification approach presented in Chapter 3, which consists of feature extraction for each curve by the RHLP model followed by MDA in the space of descriptors, is also considered.

Protocol of simulations

We consider two situations of simulated curves. The first consists of a two-class problem where the curves are generated according to two arbitrary nonlinear functions corrupted by noise. Each curve is composed of $m = 200$ points and consists of three piecewise functions such that:

- $y_t = -0.3 \times \mathbb{1}_{[0;0.25]}(t) - 5e^{-3t}\sin(2.8\pi t) \times \mathbb{1}_{]0.25;0.75]}(t) - \mathbb{1}_{]0.75;1]}(t) + \epsilon_t$ for the first class,
- $y_t = -0.1 \times \mathbb{1}_{[0;0.25]}(t) - 5e^{-3t}\sin(3\pi t) \times \mathbb{1}_{]0.25;0.75]}(t) - 0.9 \times \mathbb{1}_{[0.75;1]}(t) + \epsilon_t$ for the second class,

with $\epsilon_t \sim \mathcal{N}(0, 0.4^2)$. A sample of 100 curves is considered. Figure 4.12 shows an example of simulated curves for this situation.

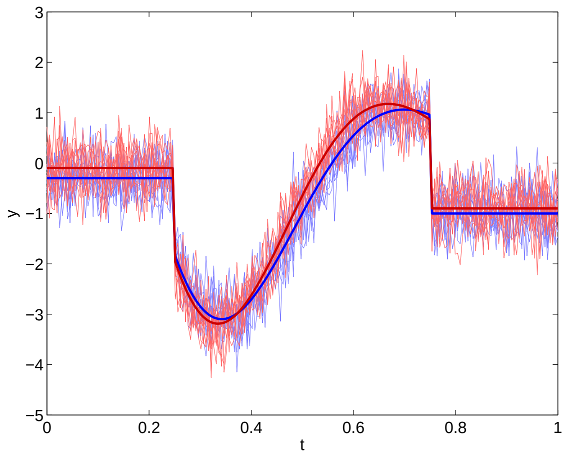


Figure 4.12: A two-class example of curves simulated according to two nonlinear functions corrupted by noise. The plot shows 10 curves from each class with the true mean functions presented in bold lines. Each curve is composed of $m = 200$ points and consists of three piecewise functions.

In the second situation, the proposed RHLP model is evaluated in terms of curve classification by considering the waveform curves of Breiman (Breiman et al., 1984).

Waveform curves of Breiman

The waveform data introduced by Breiman et al. (1984) and studied in Hastie and Tibshirani (1996); Rossi and Conan-Guez (2005a) and elsewhere consist of a three-class problem where each curve is generated as follows:

- $\mathbf{y}_1(t) = uh_1(t) + (1-u)h_2(t) + \epsilon_t$ for the class 1;
- $\mathbf{y}_2(t) = uh_2(t) + (1-u)h_3(t) + \epsilon_t$ for the class 2;
- $\mathbf{y}_3(t) = uh_1(t) + (1-u)h_3(t) + \epsilon_t$ for the class 3.

where u is a uniform random variable on $(0, 1)$,

- $h_1(t) = \max(6 - |t - 11|, 0)$;
- $h_2(t) = h_1(t - 4)$;
- $h_3(t) = h_1(t + 4)$.

and ϵ_t is a zero-mean Gaussian noise with unit standard deviation. The temporal interval considered for each curve is $[0; 20]$ with a constant period of sampling of 1 second. 500 simulated curves were drawn for each class.

Obtained results

Table 4.4 shows the average misclassification error rates and the corresponding standard deviations (in parentheses) obtained with the five approaches for the first data set. The RHLP modeling approach for a set of curves followed by the MAP classification rule provides the smallest classification error compared to the alternatives while the PWPR approach provides closely similar classification results. In practice, one observes that, for the PR model, the misclassification error rate tends to increase as the polynomial degree increases. This behavior is also concerned with the cubic spline when the PSR model includes a high number of knots. This behavior can be attributed to the phenomenon of over-fitting.

Table 4.5 shows the average classification error rates and the corresponding standard deviations (in parentheses) obtained with the five approaches for the waveform curves. We can see that, in terms of accuracy, the RHLP-MAP approach outperforms the other four methods. The difference is largest on the first data set, in particular compared to the PR and the PSR models. This is due to the fact that the class shapes for the previous data set (situation 1) present are more complex for such models, compared to the waveform curves. In addition, the error rates for the waveform data set are smaller than those obtained for situation 1 because the classes are more difficult to separate (see Figure 4.12).

Figure 4.13 and Figure 4.14 show, for the two situations, the curves estimated respectively by the four models: PR, PSR, PWPR and RHLP. We can see that, as the class-shapes are not much complex in terms of nonlinearity, all the model provide

Approach	Misclassification error rates (%)
PR-MAP ($p = 20$)	14±(8.94)
PR-MAP ($p = 10$)	9±(9.61)
PR-MAP ($p = 6$)	8.24±(7.58)
PSR-MAP (15 knots)	9±(9.61)
PSR-MAP (9 knots)	8±(7.58)
PSR-MAP (5 knots)	8±(7.58)
PWPR-MAP	7.75 ±(5.71)
RHLP-MDA	8 ±(3.54)
RHLP-MAP	7 ±(3.86)

Table 4.4: Classification results for the first situation of simulated curves. The PSR model consists of a cubic spline. The RHLP and the PWPR approaches are performed with $K = 5$ and $p = 2$.

Modeling approach	Misclassification error rates (%)
PR-MAP	2.53 ± (0.41)
PSR-MAP	2.16 ± (0.20)
PWPR-MAP	2.4 ± (0.64)
RHLP-MDA	1.83 ±(0.91)
RHLP-MAP	1.67 ±(0.84)

Table 4.5: Classification results for the waveform curves obtained with the different approaches. The polynomial regression model is performed with $p = 6$ and a cubic spline is used with 2 internal knots. The piecewise and the RHLP models are performed with $K = 2$ and $p = 3$.

similar approximation results. While, the curves estimated by the piecewise regression approach present discontinuities since they computed from a hard segmentation of the curves. The curves approximation provided with the proposed regression model is continuous due to the use of the logistic function adapted to both smooth and abrupt regime changes.

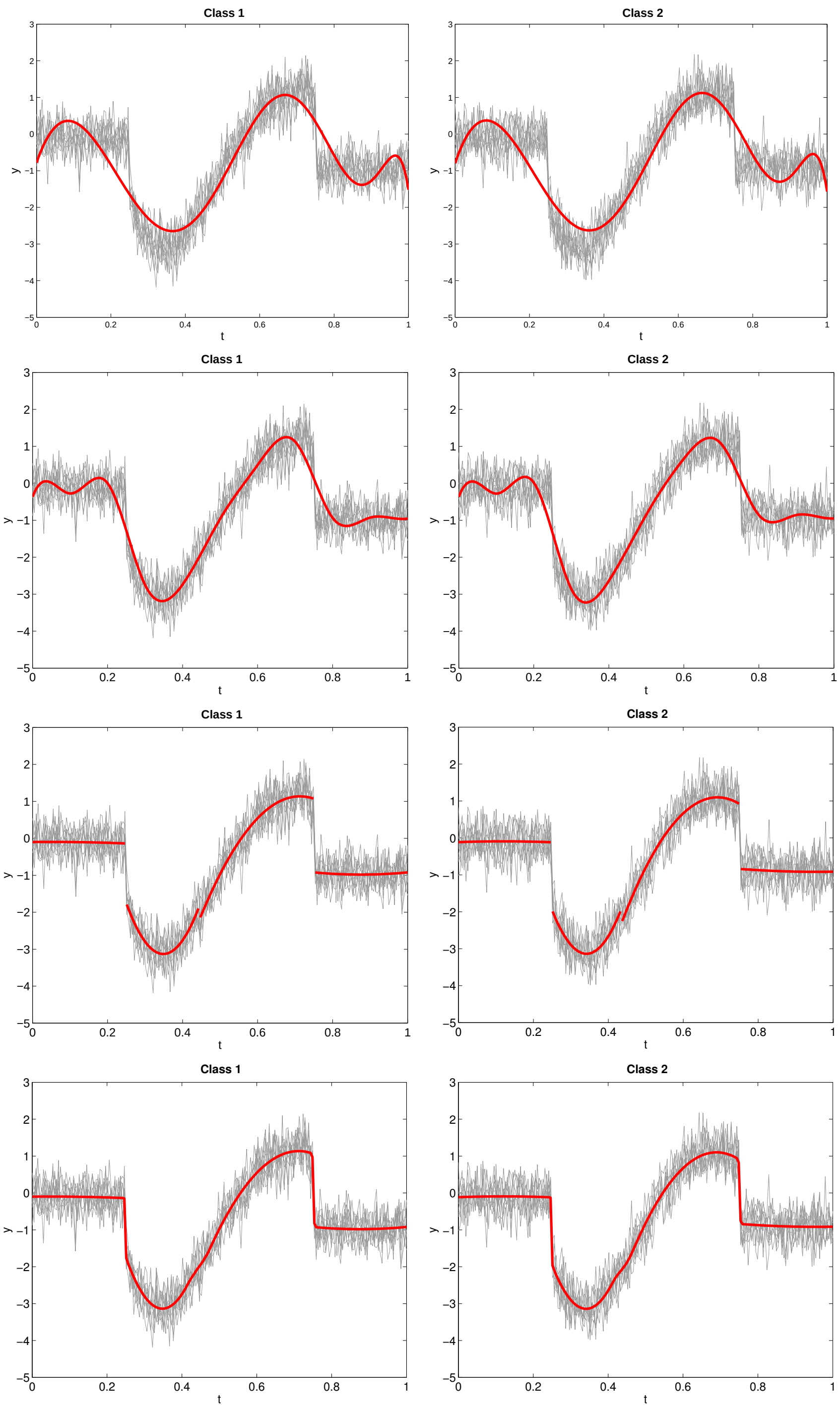


Figure 4.13: Some examples of curves corresponding to situation 1 with 10 curves per class. From top to bottom, the bold line show the estimated mean curve for each set of curves obtained with polynomial regression (PR) for $p = 15$, cubic spline regression (PSR) with 9 internal knots, piecewise polynomial regression (PWPR) with $K = 4$ and $p = 2$ and the proposed RHLP model with $K = 4$ and $p = 2$.

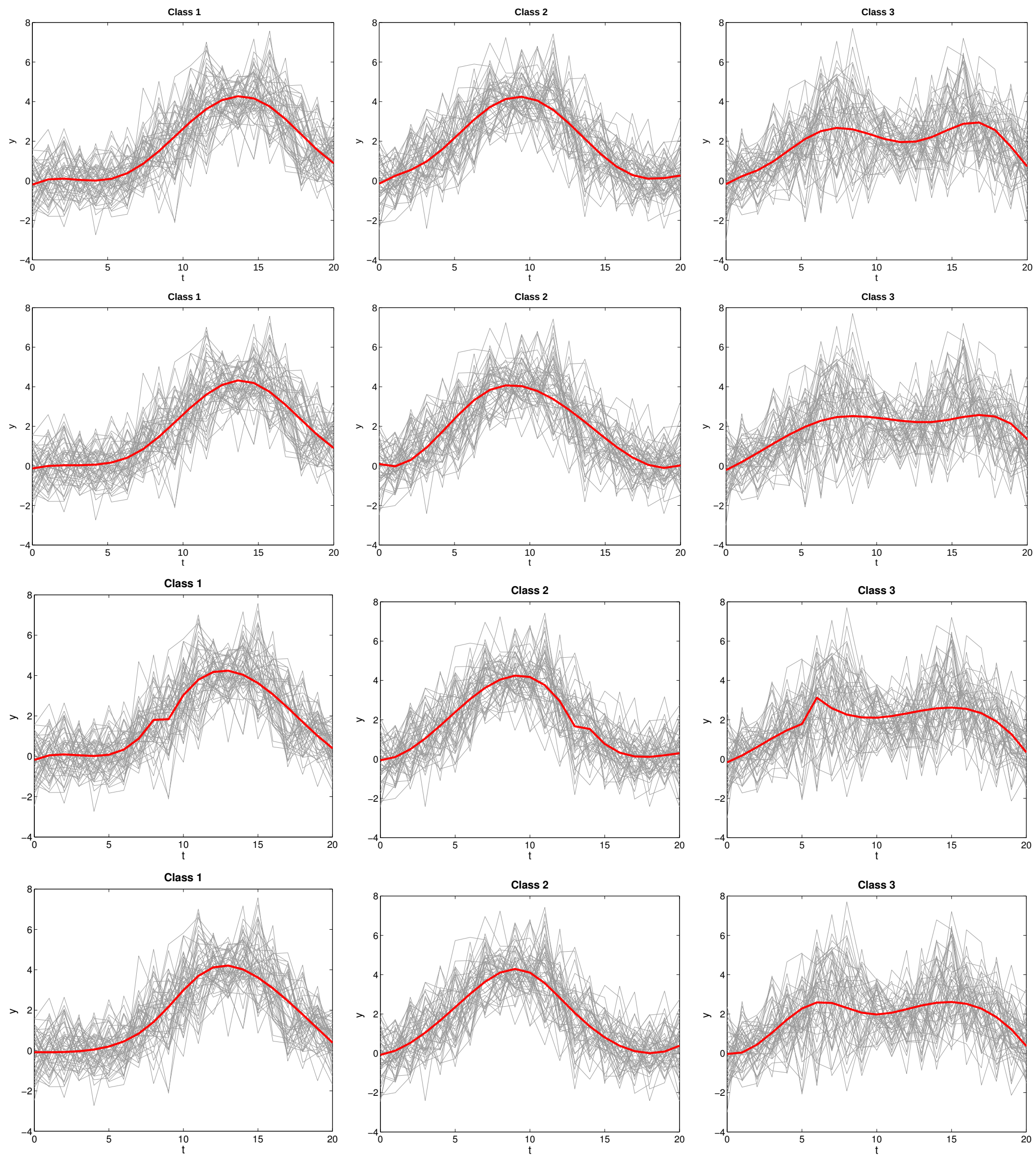


Figure 4.14: Some examples from waveform curves with 50 curves per class. From top to bottom, the bold line show the estimated mean curve for each set of curves obtained with polynomial regression (PR) for $p = 6$, cubic spline regression (PSR) with 2 internal knots, piecewise polynomial regression (PWPR) with $K = 2$ and $p = 3$ and the proposed RHLP model with $K = 2$ and $p = 3$.

4.4.5 Evaluation of the MixRHLP model in terms of curve clustering

In this section, we assess the MixRHLP model in terms for curve clustering by considering simulated curves. The first simulated example aims at illustrating the ability of the MixRHLP model for clustering curves presenting regime changes with different smoothness levels and at different temporal locations.

The simulated data consisted of curves issued from three classes which include both abrupt and smooth regime changes. The second cluster is simulated as a piecewise function and the two other clusters are simulated according to the RHLP model. Figure 4.15 shows the simulated curves.

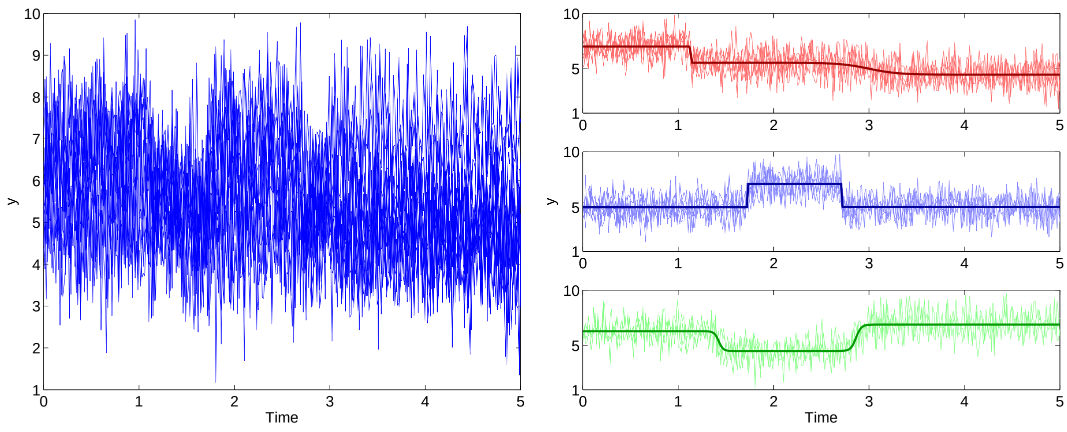


Figure 4.15: A three-class data set of simulated curves with $n = 15$, $m = 300$ and $\sigma = 1$.

In the second experiment we compare the MixRHLP model with the following alternative methods:

- the polynomial regression mixture PRM (c.f., section 2.7.1);
- the polynomial spline regression mixture PSRM (c.f., section 2.7.1);
- the polynomial piecewise regression mixture (c.f., section 2.7.3) which we abbreviate as PWPRM ;
- the mixture of HMMs (c.f., section 2.7.4) which we abbreviate as MixHMM.

The evaluation criteria are the curve classification error rate and the mean squared error between the true mean curves and the estimated curves. It is computed as:

$$\frac{1}{mR}\sum_{r=1}^{R}\sum_{j=1}^{m}(y_j^r - \hat{y}_j^r)^2$$

where y_j^r is the jth point of the mean curve of cluster r.

In the second situation, we consider two clusters of curves issued from nonlinear functions observed on [0,5] seconds. Each point of the first mean curve is computed as $y_j^1 = \sin(\pi t_j)e^{0.1t_j} + \sin(2\pi t_j)$, and the one from the second mean curve is given by $y_j^2 = \sin(\pi t_j)e^{0.1t_j} + 1.6.\sin(2\pi t_j)$. The curves are simulated with $n = 20$ and $m = 100$. The noise level is varying from 0.1 to 0.5 by step of 0.1 in order to assess the models with regards to the clusters overlap.

Figure 4.16 shows a simulated unlabeled curves and the curves labeled according to the true simulated classes.

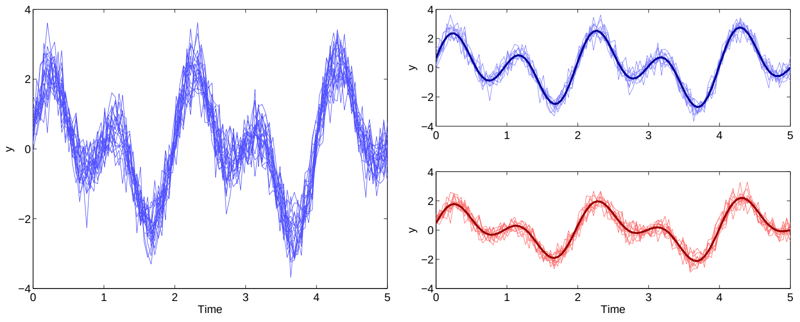


Figure 4.16: Simulated curves from two clusters (left) and the clusters labeled according to the true partition and the corresponding true mean functions (right) for situation 2.

Clustering results

Figure 4.17 shows the curves partition obtained with the CEM algorithm.

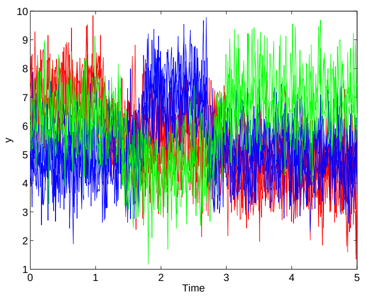


Figure 4.17: Obtained partition with the CEM algorithm for the curves shown in Figure 4.15.

Figure 4.18 clearly shows that applying the MixRHLP for clustering a set of curves presenting regime changes provides accurate results, with regard to the identified clusters, as well as for detecting the underlying regimes with each cluster. In particular, it can be seen from the logistic proportions shown in (Figure 4.18 (bottom)), that the logistic process governing each cluster of curves, is adapted for capturing both the regime transition quality and the transition locations.

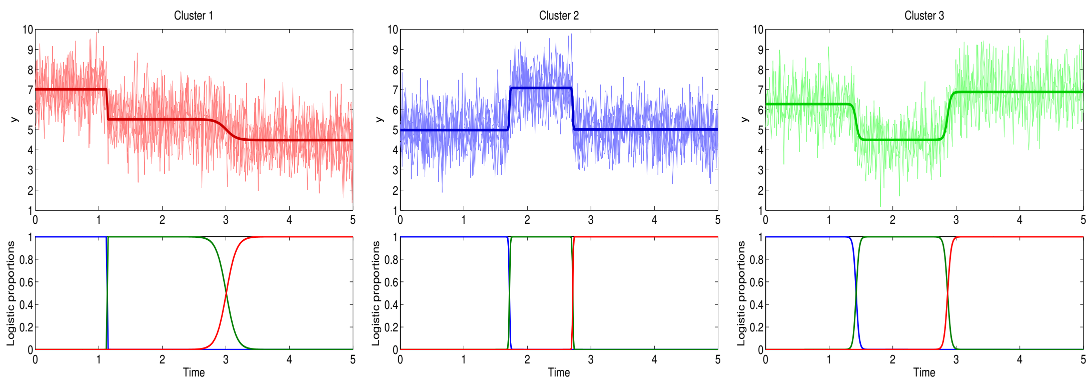


Figure 4.18: Clustering results obtained by applying the MixRHLP model with ($K = 3, p = 0$) to the set of curves presented in Figure 4.15.

For the second experiment, the polynomial regression is performed with a polynomial degree $p = 6$. The spline regression model uses continuous polynomials of degree 3 (cubic spline) and is performed with 8 internal knots.

In this case, the classification error rates equal zero for all the approaches. Figure 4.19 shows the mean squared in relation to the noise level obtained with the different clustering approaches for this situation. One can observe that the polynomial regression mixture (PRM) and the PSRM model provide less accurate results than the three other approaches. The MixRHLP model which is particularly adapted for modeling smooth curves, provides the best results for this situation, compared to the PWPRM and MixHMM model.

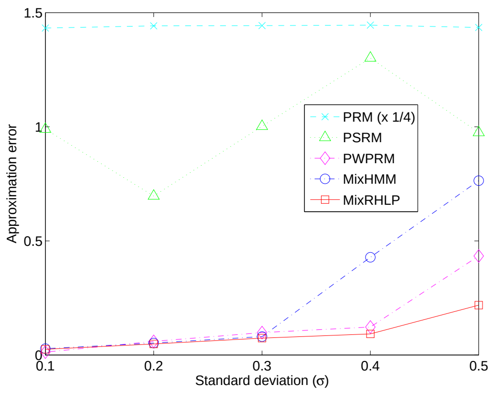


Figure 4.19: Mean Squared Error in relation to the noise level for the five curve clustering approaches.

The curves partitions and the corresponding mean functions obtained with each model, are shown in figures 4.20, 4.21, 4.22 and 4.23. It can be seen that, while the estimated clusters are quasi-identical for all the approaches, the estimated cluster models however clearly differ according to the used clustering method. The results obtained with the PSRM model are reported in Figure 4.20. In particular, it can be seen that the MixHMM model does not account well for the smoothness of the true mean curves. We note that the mean curves obtained the MixHMM model are computed as the weighted mean of the curves where the weights are both the posterior cluster probabilities and the posterior segment probabilities, that is, each point of the estimated mean curve for cluster r is given by:

$$\hat{y}_j^r = \frac{1}{\sum_{i=1}^{n}\hat{\gamma}_{ir}}\sum_{i=1}^{n}\hat{\gamma}_{ir}\sum_{k=1}^{K}\hat{\tau}_{ijrk}y_{ij}.$$

Figure 4.22 shows the clustering results, including the piecewise mean functions obtained with the MixPWPR model. Finally, Figure 4.23 presents the outputs from the MixRHLP model. One can clearly observe that the estimated cluster mean curves are very close to the original mean curves.

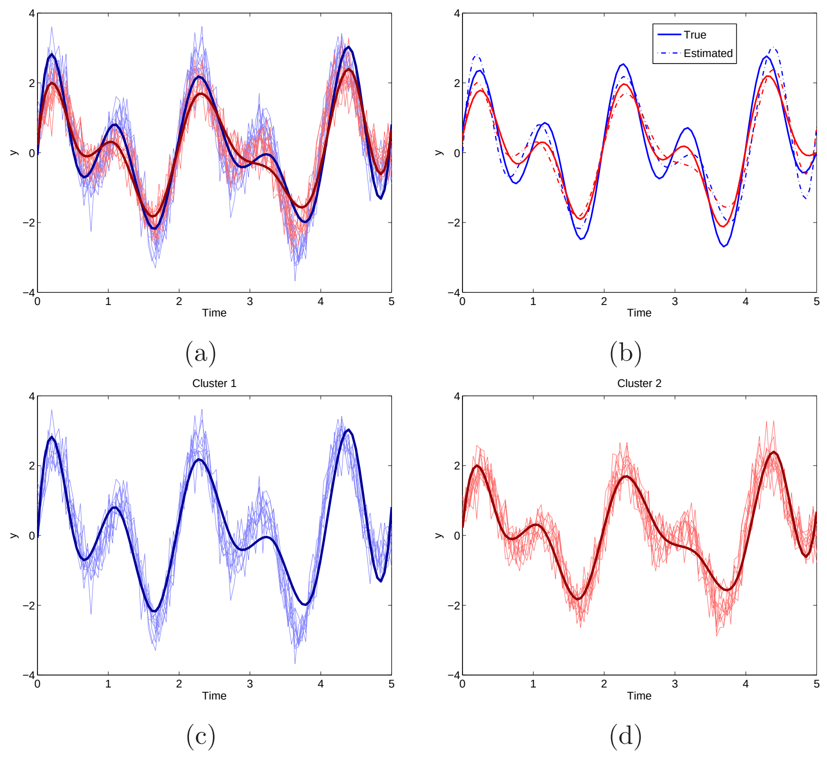


Figure 4.20: Clustering results obtained with the cubic PSRM model for the curves shown in Figure 4.16, with 8 internal uniform knots. The plot (a) shows the curves labeled according to the estimation partition and the corresponding estimated mean curve for each cluster (in bold lines). In (b) we present the true mean functions and those estimated by the PSRM model. The plots (c) and (d) show each of the two estimated clusters and the corresponding mean curves.

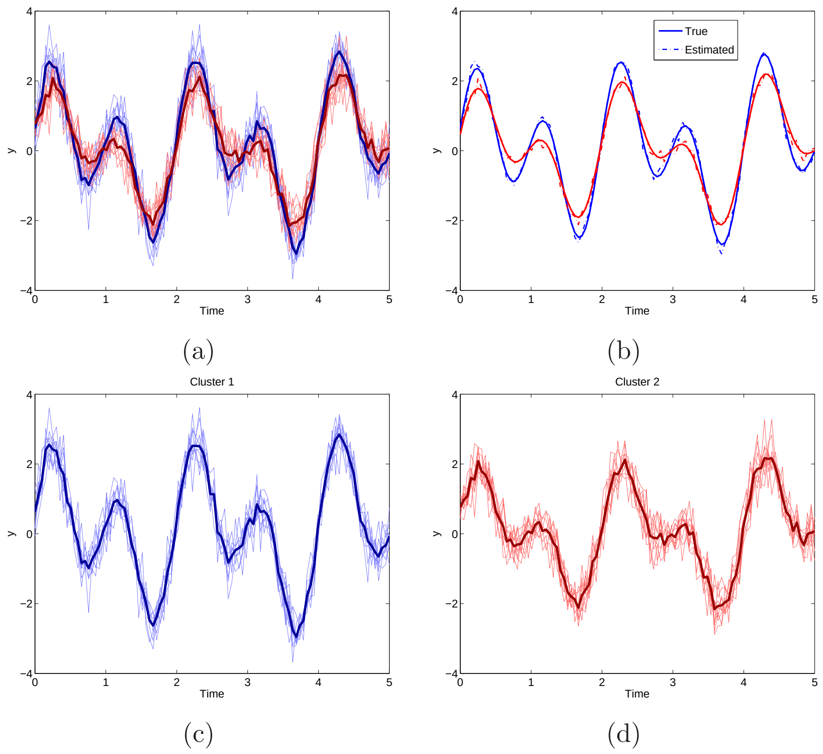


Figure 4.21: Clustering results obtained with the MixHMM model for the curves shown in Figure 4.16, with $K = 6$ and $p = 3$. The plot (a) shows the curves labeled according to the estimation partition and the corresponding estimated mean curve for each cluster (in bold lines). In (b) we present the true mean functions and those estimated by the MixHMM model. The plots (c) and (d) show each of the two estimated clusters and the corresponding mean curves.

4.5 Summary

In this chapter, we first introduced a new approach for modeling a set of curves. It consists of the RHLP model for a set of curves. The model is particularly appropriate for curves with various changes in regime. This flexibility is achieved due to the underlying logistic process. We then showed how curve classification can be performed with this model through functional linear discriminant analysis.

We further integrated the RHLP model into a mixture framework to address the problem of complex shaped sets of curves. The resulting MixRHLP model can be used for curve classification in the case of complex shaped classes using functional mixture discriminant analysis and for curve clustering.

For both the RHLP and the MixRHLP models, curve classification (or clustering) is directly performed in the space of curves, rather than in the space of curve descriptors, as in the previous chapter.

The experimental results demonstrated the benefit of the proposed approaches to

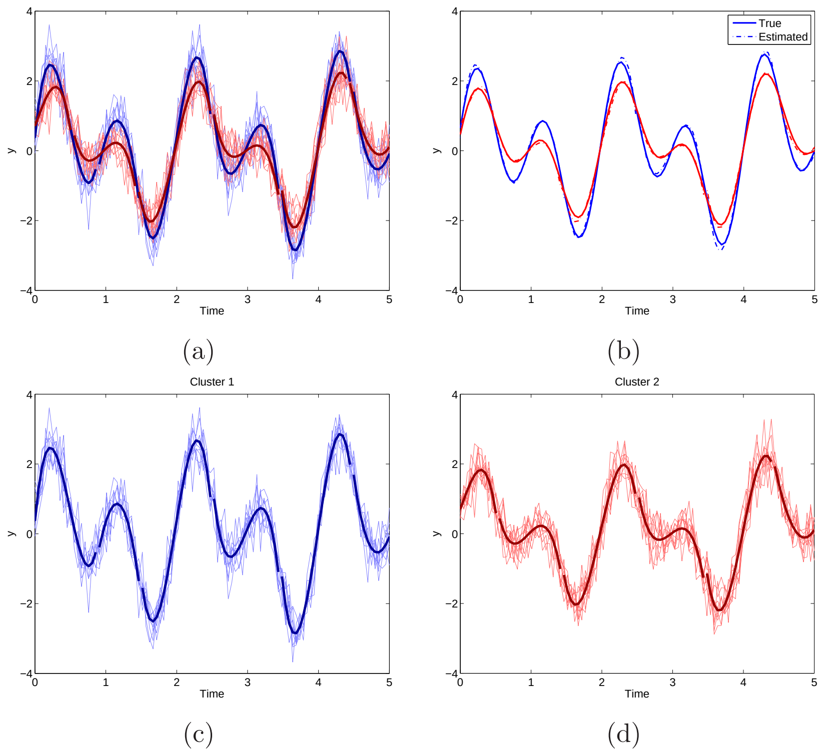
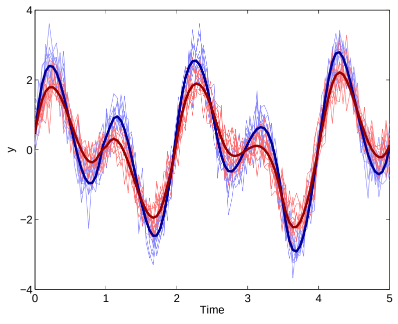


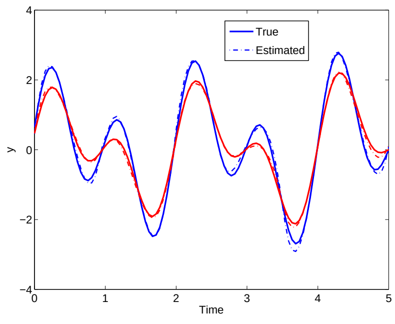
Figure 4.22: Clustering results obtained with the MixPWPR model for the curves shown in Figure 4.16, with $K = 6$ and $p = 3$. The plot (a) shows the curves labeled according to the estimation partition and the corresponding estimated piecewise mean curve for each cluster (in bold lines). In (b) we present the true mean functions and those estimated by the MixHMM model. The plots (c) and (d) show each of the two estimated clusters and the corresponding piecewise mean curves.

addressing the problem of curve modeling, classification and clustering as compared to existing alternative methods, including the piecewise polynomial regression, the polynomial spline regression and the Hidden Markov Model regression.

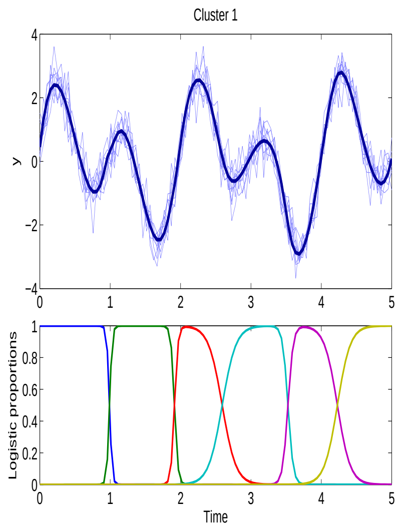
At this stage, the curves are assumed to be independent. In the next chapter, we relax this hypothesis and build models for curve sequences.



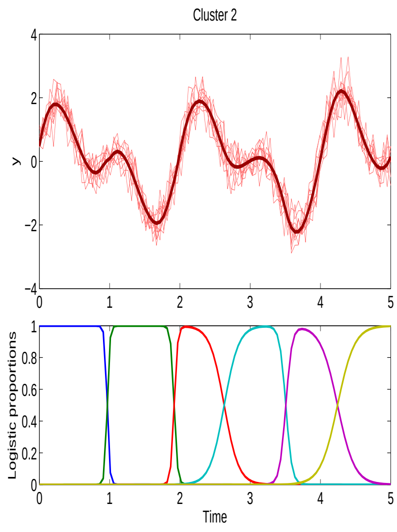
(a)



(b)



(c)



(d)

Figure 4.23: Clustering results obtained with the MixRHLP model for the curves shown in Figure 4.16, with $K = 6$ and $p = 3$. The plot (a) shows the curves labeled according to the estimation partition and the corresponding estimated mean curve for each cluster (in bold lines). In (b) we present the true mean functions and those estimated by the MixRHLP model. The plots (c) and (d) show each of the two estimated clusters, the mean curves and the corresponding logistic probabilities.

Chapter 5

Dynamical approaches for curve sequence modeling

Contents

5.1 Introduction

The previous chapters were concerned with modeling and classifying independent curves. However, in many application domains, including the railway infrastructure diagnosis, the data are generally acquired sequentially. Figure 5.1 shows a sequence of curves acquired during successive switch operations.

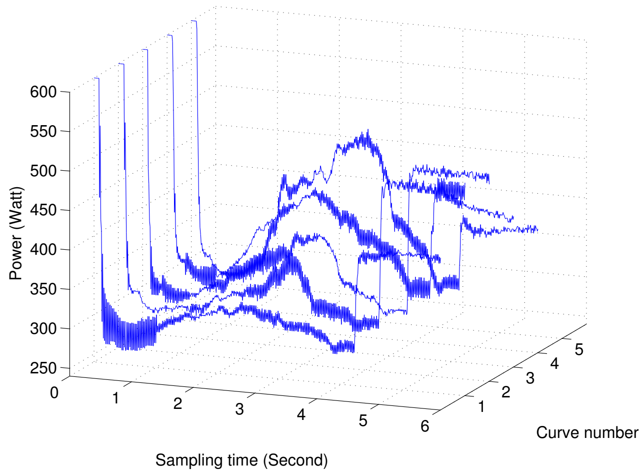


Figure 5.1: Examples of curves acquired during successive switch operations.

These sequential data can be used to learn the dynamic behavior of a system, such as is the case in condition monitoring contexts (Smyth, 1994). Indeed, because the system might be affected by operating conditions over time, a change in the operating process can be observed and detected from successive condition measurements acquired during the working process of that system. To detect non-normal events occurring during the operating process of the studied system, accurate modeling techniques are therefore needed. In this monitoring context, several probabilistic modeling approaches have been proposed, including the approach based on HMMs (Smyth, 1994) or the approach that combines an HMM and a Neural Network (Smyth, 1993).

In this chapter, we present probabilistic modeling approaches for dynamical modeling of curve sequences.

5.2 A two-steps strategy for curve sequence modeling

The approach we propose to model a curve sequence is two-fold. First, we perform feature extraction from each curve by applying the RHLP model developed in Chapter 3 (Chamroukhi et al., 2009a,c). This involves summarizing each curve in a multidimensional feature vector. Therefore, the curve sequence is converted to a sequence of multidimensional feature vectors that are denoted hereafter as $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$, where $\mathbf{y}_t$

$(t = 1, \ldots, n)$ is the d-dimensional[1] feature vector extracted from the curve observed at the time step t $(t = 1, \ldots, n)$.

Then, based on the formed sequence of multidimensional data, one can adopt probabilistic models that are generally concerned with multidimensional data sequences, such as HMMs.

The models developed in this chapter rely on a specific multivariate autoregressive model governed by a hidden discrete process that automatically controls the switching from one state to another in K states over time. Two types of processes are considered. The first approach assumes that the underlying process governing the observation sequence is a hidden logistic process, as introduced in Chapter 2. However, the logistic model described in the present chapter depends on a history of the observed variable itself rather than the time step. The resulting model is called a switching autoregressive model with a hidden logistic process, which we abbreviate as ARHLP. A hidden Markov process (i.e., hidden Markov chain) is used in the second approach; the corresponding model is a switching autoregressive HMM (ARHMM).

The learning task is performed in both a batch mode, where the observation sequence is stored in advance, and an online mode, where the data arrive one at a time. The final objective is to evaluate the operating state of the system given a new acquired observation (i.e., curve) based on the learned model and to be able to make class prediction.

This chapter is organized as follows. Section 5.3 presents the ARHLP model and its unsupervised learning technique in both an offline and an online scheme. In section 5.4, after recalling the standard ARHMM proposed by Celeux et al. (2004), we present the AR-NH-HMM based on a non-stationary modeling of the transition probabilities and describes the corresponding parameter estimation technique via the EM (Baum Welch) algorithm. Finally, the proposed approaches are evaluated in section 5.5 using an experimental study on simulated data.

5.3 Switching autoregressive model with a hidden logistic process

This section presents the proposed dynamical autoregressive model with a hidden logistic process (ARHLP). The ARHLP model is an extension of the model presented in Harrison et al. (2003), which consists of a standard multivariate autoregressive model for learning from multivariate time series, by including the latent logistic process that allows for switching between various multivariate autoregressive models. The model can be associated with the method of Wong and Li (2001) which is based on a switching autoregressive logistic model. In Wong and Li (2001), the authors developed a similar model including a hidden logistic process where the hidden discrete variable takes only two configurations (binary unsupervised classification). Other similar AR models can

[1]Note that here the dimension of the extracted feature vector $\mathbf{y}_t$ is $\nu_{\boldsymbol{\theta}}$ as seen in the previous chapters, however d will be used hereafter to simplify the notations.

be found in Carvalho and Tanner (2006, 2007) which are based on switching autoregressive logistic model. However for these approaches, the learning task is performed in a batch mode. The proposed ARHLP model includes a multinomial logistic process, for which the learning procedure is performed in a batch mode, as well as in an online mode. A similar online learning mode can be found in Ng et al. (2006).

5.3.1 The model

The observation sequence $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ is assumed to be generated by the following multivariate autoregressive model of order p, governed by a hidden logistic process:

$$\mathbf{y}_t = \sum_{a=1}^{p} \mathbf{B}^a_{z_t} \mathbf{y}_{t-a} + \mathbf{e}_t, \quad \mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{z_t}) \quad (t = p+1, \ldots, n), \tag{5.1}$$

where, for $t = 1, \ldots, p$, the initial conditions for the model are defined as:

- $z_t \sim \mathcal{M}(1, \pi_1, \ldots, \pi_K)$ such that $\pi_k \in [0, 1]$ $\forall k$ and $\sum_{k=1}^{K} \pi_k = 1$,
- $\mathbf{y}_t | z_t \sim \mathcal{N}(\cdot; \boldsymbol{\mu}_{z_t}, \mathbf{\Sigma}_{z_t})$.

The variable z_t from the underlying process $\mathbf{z} = (z_1, \ldots, z_n)$ is a latent discrete random variable which takes its values in the finite set $\mathcal{Z} = \{1, \ldots, K\}$. The variable z_t represents the unobserved class label of the sub-autoregressive model generating $\mathbf{y}_t$, $\mathbf{B}^a_{z_t}$ is the $d \times d$ matrix of autoregression parameters associated with $\mathbf{y}_{t-a}$, $\mathbf{\Sigma}_{z_t}$ is the $d \times d$ covariance matrix for the sub-autoregressive model z_t, and the variables $\boldsymbol{\epsilon}_t$ are independent random variables in $\mathbb{R}^d$ distributed according to a standard multivariate Gaussian distribution representing an additive noise. The model (5.1) can be reformulated in a vectorial form as

$$\mathbf{y}_t = \mathbf{B}^T_{z_t} \mathbf{r}_t + \mathbf{e}_t \quad (t = p+1, \ldots, n), \tag{5.2}$$

where $\mathbf{r}_t = (\mathbf{y}^T_{t-1}, \ldots, \mathbf{y}^T_{t-p})^T$ is the $(p \times d) \times 1$ dimensional vector of the p previous observations and $\mathbf{B}_{z_t} = (\mathbf{B}^1_{z_t}, \ldots, \mathbf{B}^p_{z_t})^T$ is the $(p \times d) \times d$ dimensional matrix of the K sub-autoregressive model parameters.

In this specific autoregressive model, the variable z_t controls the switching from a sub-autoregressive model to another among K sub-autoregressive models at each time step t. Thus, unlike the autoregressive model developed by Harrison et al. (2003), which assumes uniform AR model parameters over time (the parameters $(\mathbf{B}, \mathbf{\Sigma})$ are the same for the whole sequence of data), the formulation of the proposed model is based on a set of dynamical parameters, which is able to capture the dynamical (non-stationary) behavior of the process generating the data. This dynamical aspect is represented by the switching within various sub-models over time through an underlying hidden process. In this section, we assume that the underlying hidden switching process is logistic, that is the hidden variable z_t is governed by a multinomial logistic distribution in which the covariate variables consist of some short history of observed data during time.

The next section defines the probability distribution of the process $\mathbf{z} = (z_1, \ldots, z_n)$ that allows for the switching from one sub-autoregressive model to another among K autoregressive models, in the same manner as in Chapter 3.

The hidden logistic process

Let us recall that the hidden logistic process proposed in Chapter 3 for a curve depends on set of parameters and the instant at which data point of a curve is observed. For the ARHLP model we develop here, the multinomial logistic distribution depends on a set of parameters and some short history of the observed data themselves. Thus, the hidden logistic process in this case supposes that the variables z_t, given the set of p previous observations $(\mathbf{y}_{t-1}, \ldots, \mathbf{y}_{t-p})$, are generated independently according to the multinomial distribution $\mathcal{M}(1, \pi_1(\mathbf{r}_t; \mathbf{w}), \ldots, \pi_K(\mathbf{r}_t; \mathbf{w}))$ (we recall that $\mathbf{r}_t = (\mathbf{y}_{t-1}^T, \ldots, \mathbf{y}_{t-p}^T)^T$), where the conditional probability of each class (state) k $(k = 1, \ldots, K)$ is given by:

$$\pi_k(\mathbf{r}_t; \mathbf{w}) = p(z_t = k|\mathbf{r}_t; \mathbf{w}) = \frac{\exp(\boldsymbol{w}_k^T \mathbf{r}_t)}{\sum_{\ell=1}^K \exp(\boldsymbol{w}_\ell^T \mathbf{r}_t)}, \tag{5.3}$$

where $\boldsymbol{w}_k \in \mathbb{R}^{p \times d}$ is the parameter vector associated with the kth logistic component and $\mathbf{w} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K)$. Thus, given the vector $\mathbf{r}_t$, the distribution of a particular configuration of the hidden process $\mathbf{z}$ can be written as

$$\begin{aligned} p(\mathbf{z}|\mathbf{r}_{p+1}, \ldots, \mathbf{r}_n; \mathbf{w}) &= \prod_{t=1}^{p}\prod_{k=1}^{K} \pi_k^{z_{tk}} \prod_{t=p+1}^{n}\prod_{k=1}^{K} p(z_t = k|\mathbf{r}_t; \mathbf{w})^{z_{tk}} \\ &\propto \prod_{t=p+1}^{n}\prod_{k=1}^{K} \left(\frac{\exp(\boldsymbol{w}_k^T \mathbf{r}_t)}{\sum_{\ell=1}^K \exp(\boldsymbol{w}_\ell^T \mathbf{r}_t)}\right)^{z_{tk}}. \end{aligned} \tag{5.4}$$

The data generation scheme with the ARHLP model

From Equation (5.2) we can see that given the vector $\mathbf{r}_t$ comprising the p previous observations, and the sub-autoregressive model k generating $\mathbf{y}_t$, the distribution of $\mathbf{y}_t$ is a multivariate Gaussian distribution with mean $\mathbf{B}_k^T \mathbf{r}_t$ and covariance matrix $\boldsymbol{\Sigma}_k$:

$$p(\mathbf{y}_t|z_t = k, \mathbf{r}_t; \boldsymbol{\Psi}_k) = \mathcal{N}(\mathbf{y}_t; \mathbf{B}_k^T \mathbf{r}_t, \boldsymbol{\Sigma}_k) \tag{5.5}$$

where $\boldsymbol{\Psi}_k = (\mathbf{B}_k, \boldsymbol{\Sigma}_k)$. The generative scheme of an observation sequence from a fixed parameter $\boldsymbol{\Psi} = (\mathbf{w}, \boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_K)$ is therefore as follows. Given some chosen initial variables z_t $(t = 1, \ldots, p)$ and observations $\mathbf{y}_t$ $(t = 1, \ldots, p)$, for example one can chose the initial state z_t $(t = 1, \ldots, p)$ according to the multinomial distribution $\mathcal{M}(1, \pi_1, \ldots, \pi_K)$. Similarly, $\mathbf{y}_t$ $(t = 1, \ldots, p)$ can be generated according to the multivariate Gaussian distribution $\mathcal{N}(\cdot; \boldsymbol{\mu}_{z_t}, \boldsymbol{\Sigma}_{z_t})$ where $\boldsymbol{\mu}_t$ is the initial mean vector. Thus, the generative scheme is given by the following 2 steps:
For $t = p + 1, \ldots, n$,

- generate the hidden variable z_t according to the multinomial distribution $\mathcal{M}(1, \pi_1(\mathbf{r}_t; \mathbf{w}), \ldots, \pi_K(\mathbf{r}_t; \mathbf{w}))$,

- generate the observation $\mathbf{y}_t$ according to the multivariate Gaussian distribution $\mathcal{N}(\cdot; \mathbf{B}_{z_t}^T \mathbf{r}_t, \mathbf{\Sigma}_{z_t})$.

where $\mathbf{r}_t = (\mathbf{y}_{t-1}^T, \ldots, \mathbf{y}_{t-p}^T)^T$. Figure 5.2 gives a graphical representation for this model for the particular case where $p = 1$ (an ARHLP model of order 1).

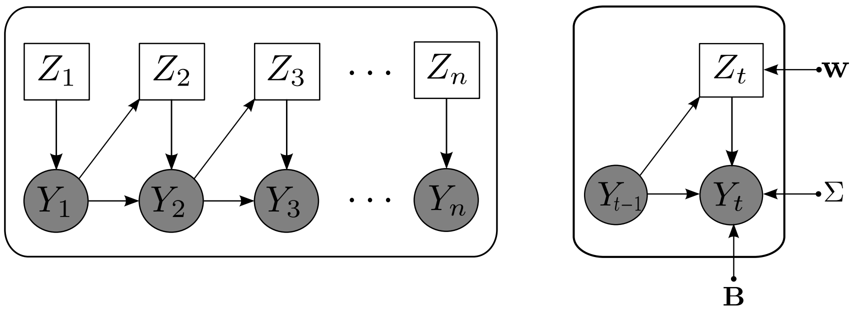


Figure 5.2: Graphical model structure for the multivariate autoregressive model with a hidden logistic process (ARHLP) for $p = 1$ (ARHLP model of order 1).

The next section is concerned with the parameter estimation for the ARHLP model in a maximum likelihood framework using the EM algorithm.

5.3.2 The batch EM algorithm for parameter estimation

It can be shown that, given the memory $\mathbf{r}_t$ of $\mathbf{y}_t$, the variable $\mathbf{y}_t$ is distributed according to the following multivariate Gaussian mixture density:

$$\begin{aligned} p(\mathbf{y}_t|\mathbf{r}_t; \mathbf{\Psi}) &= \sum_{k=1}^{K} p(z_t = k|\mathbf{r}_t; \mathbf{w}) p(\mathbf{y}_t|z_t = k, \mathbf{r}_t; \mathbf{\Psi}_k) \\ &= \sum_{k=1}^{K} \pi_k(\mathbf{r}_t; \mathbf{w}) \mathcal{N}(\mathbf{y}_t; \mathbf{B}_k^T \mathbf{r}_t, \mathbf{\Sigma}_k), \end{aligned} \tag{5.6}$$

where $\mathbf{\Psi} = (\mathbf{w}, \mathbf{B}_1, \ldots, \mathbf{B}_K, \mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_K)$ is the unknown parameter vector to be estimated by the maximum likelihood method. According to this model, the observed data $\mathbf{y}_t$ $(t = 1, \ldots, n)$ are independent given the previous observations $\mathbf{r}_t$. Assuming the specific initial condition for the initial observations $(\mathbf{y}_1, \ldots, \mathbf{y}_p)$ and the initial states $(z_1, \ldots, z_p)$, the log-likelihood of $\mathbf{\Psi}$ for the observed data $\mathbf{Y}$, given the history of observations $(\mathbf{r}_{p+1}, \ldots, \mathbf{r}_n)$ is therefore written as

$$\mathcal{L}(\mathbf{\Psi}; \mathbf{Y}) = \log \prod_{t=1}^{p} p(\mathbf{y}_t) \prod_{t=p+1}^{n} p(\mathbf{y}_t|\mathbf{r}_t; \mathbf{\Psi}) \tag{5.7}$$

from which it can be seen that up to the additive term $\sum_{t=1}^{p} \log p(\mathbf{y}_t)$[1], the observed-data log-likelihood of $\mathbf{\Psi}$ is given by

$$\mathcal{L}(\mathbf{\Psi};\mathbf{Y}) \quad \propto \quad \sum_{t=p+1}^{n} \log \sum_{k=1}^{K} \pi_k(\mathbf{r}_t;\mathbf{w})\mathcal{N}(\mathbf{y}_t;\mathbf{B}_k^T\mathbf{r}_t,\mathbf{\Sigma}_k). \tag{5.8}$$

The maximization of this log-likelihood is iteratively performed by the EM algorithm (Dempster et al., 1977; McLachlan and Krishnan, 1997). The complete-data log-likelihood for this model is written as

$$\begin{aligned}\mathcal{L}_c(\mathbf{\Psi};\mathbf{Y},\mathbf{z}) &= \log p(\mathbf{Y},\mathbf{z}|\mathbf{r}_{p+1},\ldots,\mathbf{r}_n;\mathbf{\Psi})\\ &= \log p(\mathbf{Y}|\mathbf{z},\mathbf{r}_{p+1},\ldots,\mathbf{r}_n;\mathbf{\Psi})p(\mathbf{z}|\mathbf{r}_{p+1},\ldots,\mathbf{r}_n;\mathbf{\Psi}). \end{aligned} \tag{5.9}$$

For a particular configuration of the hidden process $\mathbf{z}$, the conditional distribution of the observed sequence given the variables $(\mathbf{r}_{p+1},\ldots,\mathbf{r}_n)$, can be written as

$$p(\mathbf{Y}|\mathbf{z},\mathbf{r}_{p+1},\ldots,\mathbf{r}_n;\mathbf{\Psi})=\prod_{t=1}^{p}\prod_{k=1}^{K}\mathcal{N}(\mathbf{y}_t;\boldsymbol{\mu}_k,\mathbf{\Sigma}_k)^{z_{tk}}\prod_{t=p+1}^{n}\prod_{k=1}^{K}\mathcal{N}(\mathbf{y}_t;\mathbf{B}_k^T\mathbf{r}_t,\mathbf{\Sigma}_k)^{z_{tk}}. \tag{5.10}$$

Thus, by using (5.4) we therefore obtain the following expression of the complete-data log-likelihood:

$$\begin{aligned}\mathcal{L}_c(\mathbf{\Psi};\mathbf{Y},\mathbf{z}) \quad = \quad & \sum_{t=1}^{p}\sum_{k=1}^{K} z_{tk}\log \pi_k\mathcal{N}(\mathbf{y}_t;\boldsymbol{\mu}_k,\mathbf{\Sigma}_k) + \sum_{t=p+1}^{n}\sum_{k=1}^{K} z_{tk}\log \pi_k(\mathbf{r}_t;\mathbf{w}) +\\ & + \sum_{t=p+1}^{n}\sum_{k=1}^{K} z_{tk}\log \mathcal{N}(\mathbf{y}_t;\mathbf{B}_k^T\mathbf{r}_t,\mathbf{\Sigma}_k). \end{aligned} \tag{5.11}$$

In order to make the parallel between the batch EM algorithm, and the online EM algorithm which will be formulated later, we express the parameters' updating formulas as functions of the sufficient statistics.

Let us denote by $\mathcal{L}_c^{\mathbf{w}}(\mathbf{w};\mathbf{Y},\mathbf{z})$ and $\mathcal{L}_c^{\mathbf{\Psi}_k}(\mathbf{\Psi}_k;\mathbf{Y},\mathbf{z})$ the terms in (5.11) that depend on the parameters $\mathbf{w}$ and $\mathbf{\Psi}_k$ respectively, so that we have[2]:

$$\mathcal{L}_c(\mathbf{\Psi};\mathbf{Y},\mathbf{z}) = \mathcal{L}_c^{\mathbf{w}}(\mathbf{w};\mathbf{Y},\mathbf{z}) + \sum_{k=1}^{K}\mathcal{L}_c^{\mathbf{\Psi}_k}(\mathbf{\Psi}_k;\mathbf{Y},\mathbf{z}). \tag{5.12}$$

Hence, from (A.8) we have the following expression of $\mathcal{L}_c^{\mathbf{\Psi}_k}(\mathbf{\Psi}_k;\mathbf{Y},\mathbf{z})$:

$$\begin{aligned}\mathcal{L}_c^{\mathbf{\Psi}_k}(\mathbf{\Psi}_k;\mathbf{Y},\mathbf{z}) = \frac{-1}{2}\mathrm{trace}\Big[\sum_{t=p+1}^{n} z_{tk}\mathbf{y}_t\mathbf{y}_t^T\mathbf{\Sigma}_k^{-1} - 2\sum_{t=p+1}^{n} z_{tk}\mathbf{y}_t\mathbf{r}_t^T\mathbf{B}_k\mathbf{\Sigma}_k^{-1} + \sum_{t=p+1}^{n} z_{tk}\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\mathbf{\Sigma}_k^{-1}\mathbf{B}_k^T\Big]\\ -\frac{1}{2}\sum_{t=p+1}^{n} z_{tk}\log|\mathbf{\Sigma}_k| - \frac{d}{2}\sum_{t=p+1}^{n} z_{tk}\log 2\pi. \end{aligned} \tag{5.13}$$

[1]Note that omitting this additive term does not affect the parameter estimation since we have an autoregressive recursion only for $t = p+1,\ldots,n$.

[2]Notice that in what follows, we will omit the first constant term in (5.11) as it is not included in the learning process.

From (5.13) we can see that the complete-data log-likelihood for the parameters $\boldsymbol{\Psi}_k = (\mathbf{B}_k, \boldsymbol{\Sigma}_k)$ depends on the data only through the following sufficient statistics:

$$\begin{aligned}
\boldsymbol{T}_k &= \sum_{t=p+1}^{n} z_{tk}, \\
\boldsymbol{T}_{rr,k} &= \sum_{t=p+1}^{n} z_{tk}\mathbf{r}_t\mathbf{r}_t^T, \\
\boldsymbol{T}_{yr,k} &= \sum_{t=p+1}^{n} z_{tk}\mathbf{r}_t\mathbf{y}_t^T, \\
\boldsymbol{T}_{yy,k} &= \sum_{t=p+1}^{n} z_{tk}\mathbf{y}_t\mathbf{y}_t^T,
\end{aligned} \tag{5.14}$$

where $\boldsymbol{T}_{rr,k}$ and $\boldsymbol{T}_{yr,k}$ are sufficient statistics for $\mathbf{B}_k$, and $\boldsymbol{T}_k$, $\boldsymbol{T}_{yr,k}$ and $\boldsymbol{T}_{yy,k}$ are sufficient statistics for $\boldsymbol{\Sigma}_k$ with $k \in \{1, \ldots, K\}$. The updating formulas for $\boldsymbol{\Psi}_k$ will then be reformulated based on these sufficient statistics. In addition, the optimization of the function

$$\mathcal{L}_c^{\mathbf{w}}(\mathbf{w}; \mathbf{Y}, \mathbf{z}) = \sum_{t=p+1}^{n}\sum_{k=1}^{K} z_{tk} \log \pi_k(\mathbf{r}_t; \mathbf{w}) \tag{5.15}$$

with respect to $\mathbf{w}$ is a multinomial logistic regression problem, so that if the hidden indicator variables z_{tk} were observed, the estimation of $\mathbf{w}$ would be performed using the IRLS algorithm as follows:

$$\mathbf{w}^{(l+1)} = \mathbf{w}^{(l)} - \Big[\frac{\partial^2 \mathcal{L}_c^{\mathbf{w}}(\mathbf{w}; \mathbf{Y}, \mathbf{z})}{\partial \mathbf{w} \partial \mathbf{w}^T}\Big]^{-1}_{\mathbf{w}=\mathbf{w}^{(l)}} \frac{\partial \mathcal{L}_c^{\mathbf{w}}(\mathbf{w}; \mathbf{Y}, \mathbf{z})}{\partial \mathbf{w}}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}}. \tag{5.16}$$

Thus, let us define the following statistics

$$\boldsymbol{T}_{rr,w}^{(l)} = -\frac{\partial^2 \mathcal{L}_c^{\mathbf{w}}(\mathbf{w}; \mathbf{Y}, \mathbf{z})}{\partial \mathbf{w} \partial \mathbf{w}^T}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}}, \text{ and } \boldsymbol{T}_{rw}^{(l)} = \frac{\partial \mathcal{L}_c^{\mathbf{w}}(\mathbf{w}; \mathbf{Y}, \mathbf{z})}{\partial \mathbf{w}}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} \tag{5.17}$$

which correspond to the Hessian and the gradient of $\mathcal{L}_c^{\mathbf{w}}(\mathbf{w}; \mathbf{Y}, \mathbf{z})$ respectively, and are calculated in a similar way as in (A.29) and (A.25) as follows:

$$\begin{aligned}
\boldsymbol{T}_{rr,w}^{(l)} &= \sum_{t=p+1}^{n} \big(\boldsymbol{\Pi}(\mathbf{r}_t; \mathbf{w}^{(l)}) - \boldsymbol{\pi}(\mathbf{r}_t; \mathbf{w}^{(l)})\boldsymbol{\pi}^T(\mathbf{r}_t; \mathbf{w}^{(l)})\big) \otimes \mathbf{r}_t\mathbf{r}_t^T, \\
\boldsymbol{T}_{rw}^{(l)} &= \sum_{t=p+1}^{n} \big(\boldsymbol{z}_t - \boldsymbol{\pi}(\mathbf{r}_t; \mathbf{w}^{(l)})\big) \otimes \mathbf{r}_t
\end{aligned} \tag{5.18}$$

where $\boldsymbol{\pi}(\mathbf{r}_t; \mathbf{w}) = (\pi_1(\mathbf{r}_t; \mathbf{w}), \ldots, \pi_{K-1}(\mathbf{r}_t; \mathbf{w}))^T$ is the vector of the $K-1$ logistic probabilities at the time step t, $\boldsymbol{\Pi}(\mathbf{r}_t; \mathbf{w}) = \text{diag}(\boldsymbol{\pi}(\mathbf{r}_t; \mathbf{w}))$ is a $(K-1) \times (K-1)$ diagonal matrix, $\boldsymbol{z}_t = (z_{t1}, \ldots, z_{t,K-1})^T$ is the vector of class indicator variables and $\otimes$ denotes the Kronecker matrix product. Thus the update formula for $\mathbf{w}$ (5.16) takes the form:

$$\mathbf{w}^{(l+1)} = \mathbf{w}^{(l)} + \big[\boldsymbol{T}_{rr,w}^{(l)}\big]^{-1}\boldsymbol{T}_{rw}^{(l)}. \tag{5.19}$$

While the data are incomplete because the variables $\mathbf{z}$ are hidden, the EM algorithm provides an elegant framework to deal with this problem of missing information. The maximization of the observed-data log-likelihood is therefore iteratively performed by alternating between the expectation step in which we compute the conditional expectation of the statistics $\boldsymbol{T}_{rr,w}^{(l)}$ and $\boldsymbol{T}_{rw}^{(l)}$, and the M-step in which the model parameters are updated based on the expected sufficient statistics. The next section gives the details of the batch EM algorithm when the observation sequence is stored before the learning proceeds.

The batch EM algorithm for the ARHLP model

The EM algorithm for the ARHLP model starts with an initial parameter $\boldsymbol{\Psi}^{(0)}$ and alternates between the two following steps until convergence:

E-step: This step consists of computing the expectation of the complete-data log-likelihood $\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{z})$ conditionally on the observed sequence $\mathbf{Y}$ and the vectors containing the previous observations $(\mathbf{r}_{p+1},\ldots,\mathbf{r}_n)$ given the current estimation $\boldsymbol{\Psi}^{(q)}$ of the parameter $\boldsymbol{\Psi}$. This expected complete-data log-likelihood is given by:

$$\begin{aligned}
Q(\boldsymbol{\Psi},\boldsymbol{\Psi}^{(q)}) &= \mathbb{E}\Big[\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{z})|\mathbf{Y},\mathbf{r}_{p+1},\ldots,\mathbf{r}_n;\boldsymbol{\Psi}^{(q)}\Big]\\
&= \sum_{t=p+1}^{n}\sum_{k=1}^{K}\mathbb{E}[z_{tk}|\mathbf{Y},\mathbf{r}_{p+1},\ldots,\mathbf{r}_n;\boldsymbol{\Psi}^{(q)})]\log\Big[\pi_k(\mathbf{r}_t;\mathbf{w})\mathcal{N}(\mathbf{y}_t;\mathbf{B}_k^T\mathbf{r}_t,\boldsymbol{\Sigma}_k)\Big]\\
&= \sum_{t=p+1}^{n}\sum_{k=1}^{K}\tau_{tk}^{(q)}\log\pi_k(\mathbf{r}_t;\mathbf{w}) + \sum_{t=p+1}^{n}\sum_{k=1}^{K}\tau_{tk}^{(q)}\mathcal{N}(\mathbf{y}_t;\mathbf{B}_k^T\mathbf{r}_t,\boldsymbol{\Sigma}_k)\Big],
\end{aligned} \tag{5.20}$$

where

$$\begin{aligned}
\tau_{tk}^{(q)} &= \mathbb{E}[z_{tk}|\mathbf{Y},\mathbf{r}_{p+1},\ldots,\mathbf{r}_n;\boldsymbol{\Psi}^{(q)})]\\
&= p(z_t=k|\mathbf{y}_t,\mathbf{r}_t;\boldsymbol{\Psi}^{(q)})\\
&= \frac{\pi_k(\mathbf{r}_t;\mathbf{w}^{(q)})\mathcal{N}(\mathbf{y}_t;\mathbf{B}_k^{T(q)}\mathbf{r}_t,\boldsymbol{\Sigma}_k^{(q)})}{\sum_{\ell=1}^K\pi_\ell(\mathbf{r}_t;\mathbf{w}^{(q)})\mathcal{N}(\mathbf{y}_t;\mathbf{B}_\ell^{T(q)}\mathbf{r}_t,\boldsymbol{\Sigma}_\ell^{(q)})}
\end{aligned} \tag{5.21}$$

is the posterior probability that $\mathbf{y}_t$ originates from the kth sub-autoregressive model. Note that in (5.21) we used the conditional independence assumption of the model which implies that the current state depends only on the previous observations $\mathbf{r}_t$ and, conditionally on the current state, the current observation depends only on the previous p observations $\mathbf{r}_t$ $(t>p)$. Therefore, this step consists of computing the posterior probabilities $\tau_{tk}^{(q)}$.

M-step: In this step, the value of the parameter $\boldsymbol{\Psi}$ is updated by computing the parameter $\boldsymbol{\Psi}^{(q+1)}$ which maximizes the function $Q(\boldsymbol{\Psi},\boldsymbol{\Psi}^{(q)})$ with respect to $\boldsymbol{\Psi}$:

$$\boldsymbol{\Psi}^{(q+1)} = \arg\max_{\boldsymbol{\Psi}\in\boldsymbol{\Omega}} Q(\boldsymbol{\Psi},\boldsymbol{\Psi}^{(q)}), \tag{5.22}$$

$\mathbf{\Omega}$ being the parameter space.
From Equation (5.20), the Q-function can be decomposed as

$$Q(\mathbf{\Psi}, \mathbf{\Psi}^{(q)}) = Q_{\mathbf{w}}(\mathbf{w}, \mathbf{\Psi}^{(q)}) + \sum_{k=1}^{K} Q_{\mathbf{\Psi}_k}(\mathbf{\Psi}_k, \mathbf{\Psi}^{(q)}) \tag{5.23}$$

where

$$Q_{\mathbf{w}}(\mathbf{w}, \mathbf{\Psi}^{(q)}) \;=\; \sum_{t=p+1}^{n} \sum_{k=1}^{K} \tau_{tk}^{(q)} \log \pi_k(\mathbf{r}_t; \mathbf{w}) \tag{5.24}$$

and

$$Q_{\mathbf{\Psi}_k}(\mathbf{\Psi}_k, \mathbf{\Psi}^{(q)}) = \sum_{t=p+1}^{n} \tau_{tk}^{(q)} \log \mathcal{N}(\mathbf{y}_t; \mathbf{B}_k^T \mathbf{r}_t, \mathbf{\Sigma}_k) \tag{5.25}$$

for $k = 1, \ldots, K$. The maximization of $Q(\mathbf{\Psi}, \mathbf{\Psi}^{(q)})$ w.r.t $\mathbf{\Psi}$ can therefore be performed by separately maximizing $Q_{\mathbf{w}}(\mathbf{w}, \mathbf{\Psi}^{(q)})$ w.r.t $\mathbf{w}$ and $Q_{\mathbf{\Psi}_k}(\mathbf{\Psi}_k, \mathbf{\Psi}^{(q)})$ w.r.t $\mathbf{\Psi}_k$ for $k = 1, \ldots, K$.

The maximization of $Q_{\mathbf{\Psi}_k}$ with respect to $\mathbf{B}_k$ consists of analytically solving a K weighted least squares problem. Consider the terms in Equation (5.25) which depend on $\mathbf{B}_k$. Taking the derivative of the resulting function (c.f, Appenidx A.9) w.r.t $\mathbf{B}_k$ and setting it to zero yields:

$$\mathbf{B}_k^{(q+1)} = \Big[\sum_{t=p+1}^{n} \tau_{tk}^{(q)} \mathbf{r}_t \mathbf{r}_t^T \Big]^{-1} \sum_{t=p+1}^{n} \tau_{tk}^{(q)} \mathbf{r}_t \mathbf{y}_t^T \tag{5.26}$$

which corresponds to the solution of a weighted least squares problem and can be written in a matrix form as

$$\mathbf{B}_k^{(q+1)} = (\mathbf{X}^T \mathbf{W}_k^{(q)} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W}_k^{(q)} \mathbf{Y} \tag{5.27}$$

where $\mathbf{X} = (\mathbf{r}_{p+1}, \ldots, \mathbf{r}_n)^T$ is the $(n-p) \times (p \times d)$ matrix containing the p previous observations at each time step t $(t = p+1, \ldots, n)$ and $\mathbf{W}_k^{(q)}$ is the $(n-p) \times (n-p)$ diagonal matrix whose diagonal elements are the posterior probabilities of the kth sub-autoregressive model $(\tau_{p+1,k}^{(q)}, \ldots, \tau_{nk}^{(q)})$.

Similarly, to maximize $Q_{\mathbf{\Psi}_k}$ with respect to $\mathbf{\Sigma}_k$, let us collect the terms in (5.25) that are functions of $\mathbf{\Sigma}_k$ (see Appendix A.10). Taking the derivative of the resulting function and setting it to zero yields:

$$\mathbf{\Sigma}_k^{(q+1)} = \frac{\sum_{t=p+1}^{n} \tau_{tk}^{(q)} \mathbf{y}_t \mathbf{y}_t^T - \mathbf{B}_k^{(q+1)T} \sum_{t=p+1}^{n} \tau_{tk}^{(q)} \mathbf{r}_t \mathbf{y}_t^T}{\sum_{t=p+1}^{n} \tau_{tk}^{(q)}}. \tag{5.28}$$

The calculation details for (5.26) and (5.28) are given in Appendix A.9 and Appendix A.10. Note that the update formula of $\mathbf{\Sigma}_k$ is the same as in the case of the estimation of the covariance matrix in a multivariate Gaussian mixture model:

$$\mathbf{\Sigma}_k^{(q+1)} = \frac{1}{\sum_{k=1}^{K} \tau_{tk}^{(q)}} \sum_{t=p+1}^{n} \tau_{tk}^{(q)} (\mathbf{y}_t - \mathbf{B}_k^{T(q+1)} \mathbf{r}_t)^T (\mathbf{y}_t - \mathbf{B}_k^{T(q+1)} \mathbf{r}_t), \tag{5.29}$$

which can be written in a matrix form as

$$\mathbf{\Sigma}_k^{(q+1)} = \frac{1}{\sum_{k=1}^{K} \tau_{tk}^{(q)}} (\mathbf{Y} - \mathbf{X}\mathbf{B}_k^{(q+1)})^T \mathbf{W}^{(q)} (\mathbf{Y} - \mathbf{X}\mathbf{B}_k^{(q+1)}). \tag{5.30}$$

However, Equation (5.28) is used because it shows clearly the sufficient statistics $\boldsymbol{T}_{yy,k}$, $\boldsymbol{T}_{yr,k}$ and $\boldsymbol{T}_k$ in the update formula of $\mathbf{\Sigma}_k$.

From these update formulas, it can be seen that the parameter updates are based on the current conditional expected sufficient statistics defined by:

$$\begin{aligned}
\boldsymbol{S}_k^{(q)} &= \mathbb{E}[\boldsymbol{T}_k|\mathbf{Y}, \mathbf{r}_{p+1}, \ldots, \mathbf{r}_n; \mathbf{\Psi}^{(q)}] = \sum_{t=p+1}^{n} \tau_{tk}^{(q)}, \\
\boldsymbol{S}_{rr,k}^{(q)} &= \mathbb{E}[\boldsymbol{T}_{rr,k}|\mathbf{Y}, \mathbf{r}_{p+1}, \ldots, \mathbf{r}_n; \mathbf{\Psi}^{(q)}] = \sum_{t=p+1}^{n} \tau_{tk}^{(q)} \mathbf{r}_t \mathbf{r}_t^T, \\
\boldsymbol{S}_{yr,k}^{(q)} &= \mathbb{E}[\boldsymbol{T}_{yr,k}|\mathbf{Y}, \mathbf{r}_{p+1}, \ldots, \mathbf{r}_n; \mathbf{\Psi}^{(q)}] = \sum_{t=p+1}^{n} \tau_{tk}^{(q)} \mathbf{r}_t \mathbf{y}_t^T, \\
\boldsymbol{S}_{yy,k}^{(q)} &= \mathbb{E}[\boldsymbol{T}_{yy,k}|\mathbf{Y}, \mathbf{r}_{p+1}, \ldots, \mathbf{r}_n; \mathbf{\Psi}^{(q)}] = \sum_{t=p+1}^{n} \tau_{tk}^{(q)} \mathbf{y}_t \mathbf{y}_t^T
\end{aligned} \tag{5.31}$$

such that we have

$$\mathbf{B}_k^{(q+1)} = \left[\boldsymbol{S}_{rr,k}^{(q)}\right]^{-1} \boldsymbol{S}_{yr,k}^{(q)} \tag{5.32}$$

and

$$\mathbf{\Sigma}_k^{(q+1)} = \frac{\boldsymbol{S}_{yy,k}^{(q)} - \mathbf{B}_k^{T(q+1)} \boldsymbol{S}_{yr,k}^{(q)}}{\boldsymbol{S}_k^{(q)}}. \tag{5.33}$$

Finally, the maximization of the function $Q_{\mathbf{w}}(\mathbf{w}, \mathbf{\Psi}^{(q)})$ given by Equation (5.24) with respect to $\mathbf{w}$ is a multinomial weighted logistic regression problem which can be iteratively solved by the IRLS algorithm. The IRLS algorithm provides the parameter update $\mathbf{w}^{(q+1)}$. In this case, a single update at iteration l of the IRLS algorithm is given by:

$$\mathbf{w}^{(l+1)} = \mathbf{w}^{(l)} - \left[\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w}, \mathbf{\Psi}^{(q)})}{\partial \mathbf{w} \partial \mathbf{w}^T}\right]^{-1}_{\mathbf{w}=\mathbf{w}^{(l)}} \frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \mathbf{\Psi}^{(q)})}{\partial \mathbf{w}}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}}. \tag{5.34}$$

The Hessian and the gradient of $Q_{\mathbf{w}}(\mathbf{w}, \mathbf{\Psi}^{(q)})$ evaluated at $\mathbf{w} = \mathbf{w}^{(l)}$ are computed similarly as in equations (A.29) and (A.25), and are given in a matrix form by:

$$\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w}, \mathbf{\Psi}^{(q)})}{\partial \mathbf{w} \partial \mathbf{w}^T}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} = -\sum_{t=p+1}^{n} \Big(\mathbf{\Pi}(\mathbf{r}_t; \mathbf{w}^{(l)}) - \boldsymbol{\pi}(\mathbf{r}_t; \mathbf{w}^{(l)}) \boldsymbol{\pi}^T(\mathbf{r}_t; \mathbf{w}^{(l)})\Big) \otimes \mathbf{r}_t \mathbf{r}_t^T \tag{5.35}$$

and

$$\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \mathbf{\Psi}^{(q)})}{\partial \mathbf{w}}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} = \sum_{t=p+1}^{n} \big(\boldsymbol{\tau}_t^{(q)} - \boldsymbol{\pi}(\mathbf{r}_t; \mathbf{w}^{(l)})\big) \otimes \mathbf{r}_t \tag{5.36}$$

where $\boldsymbol{\tau}_t^{(q)} = (\tau_{t1}^{(q)}, \ldots, \tau_{t,K-1}^{(q)})^T$. Thus, by considering the following expected statistics:

$$\begin{aligned}\boldsymbol{S}_{rr,w}^{(q,l)} &= \mathbb{E}[\boldsymbol{T}_{rr,w}^{(l)}|\mathbf{Y}, \boldsymbol{\Psi}^{(q)}] = \sum_{t=p+1}^{n} \Big(\boldsymbol{\Pi}(\mathbf{w}^{(l)}) - \boldsymbol{\pi}(\mathbf{r}_t; \mathbf{w}^{(l)})\boldsymbol{\pi}^T(\mathbf{r}_t; \mathbf{w}^{(l)})\Big) \otimes \mathbf{r}_t\mathbf{r}_t^T, \\ \boldsymbol{S}_{rw}^{(q,l)} &= \mathbb{E}[\boldsymbol{T}_{rw}^{(l)}|\mathbf{Y}, \boldsymbol{\Psi}^{(q)}] = \sum_{t=p+1}^{n} \big(\boldsymbol{\tau}_t^{(q)} - \boldsymbol{\pi}(\mathbf{r}_t; \mathbf{w}^{(l)})\big) \otimes \mathbf{r}_t, \end{aligned} \tag{5.37}$$

we can see that the iterative update formula of $\mathbf{w}$ (5.34) takes the form:

$$\mathbf{w}^{(l+1)} = \mathbf{w}^{(l)} + \Big[\boldsymbol{S}_{rr,w}^{(q,l)}\Big]^{-1} \boldsymbol{S}_{rw}^{(q,l)}. \tag{5.38}$$

This batch EM algorithm for the ARHLP model is summarized by the pseudo code 7.

Algorithm 7 Pseudo code of the batch EM algorithm for the ARHLP model.

Inputs: training set of n feature vectors $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$, the number of operating states K and the order p of the autoregressive mode.

```
1: Initialize: $\boldsymbol{\Psi}^{(0)} = (\mathbf{w}^{(0)}, \boldsymbol{\Psi}_1^{(0)}, \ldots, \boldsymbol{\Psi}_K^{(0)})$ with $\boldsymbol{\Psi}_k^{(0)} = (\mathbf{B}_k^{(0)}, \boldsymbol{\Sigma}_K^{(0)})$ for $k = 1, \ldots, K$.
2: fix a threshold $\epsilon > 0$
3: set $q \leftarrow 0$ (EM iteration)
4: while increment in log-likelihood $> \epsilon$ do
5:    E-Step:
6:    for $k = 1, \ldots, K$ do
7:       compute $\tau_{tk}^{(q)}$ for $t = p+1, \ldots, n$ using Equation (5.21)
8:    end for
9:    M-Step:
10:   for $k = 1, \ldots, K$ do
11:      compute $\mathbf{B}_k^{(q+1)}$ using Equation (5.26)
12:      compute $\boldsymbol{\Sigma}_k^{(q+1)}$ using Equation (5.28)
13:      IRLS:
14:      Initialize: set $\mathbf{w}^{(l)} = \mathbf{w}^{(q)}$
15:      set a threshold $\delta > 0$
16:      $l \leftarrow 0$ (IRLS iteration)
17:      while increment in $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\Psi}^{(q)}) > \delta$ do
18:         compute $\mathbf{w}^{(l+1)}$ using Equation (5.34)
19:         $l \leftarrow l + 1$
20:      end while
21:      $\mathbf{w}^{(q+1)} \leftarrow \mathbf{w}^{(l)}$
22:      $q \leftarrow q + 1$
23:   end for
24: end while
25: $\hat{\boldsymbol{\Psi}} = \boldsymbol{\Psi}^{(q)}$
26: $\hat{\tau}_{tk} = \tau_{tk}^{(q)}$ for $(t = p+1, \ldots, n)$ and $k = 1, \ldots, K$
```

Outputs: $\hat{\boldsymbol{\Psi}} = (\hat{\mathbf{w}}, \hat{\boldsymbol{\Psi}}_1, \ldots, \hat{\boldsymbol{\Psi}}_K)$; $\hat{\tau}_{tk}$ for $(t = p+1, \ldots, n)$ and $k = 1, \ldots, K$

In the next two paragraphs we will show how the ARHLP model can be used for state estimation and prediction.

A posteriori state estimation

Suppose we have estimated the parameters of the different states (classes) from a training observation sequence observed up to time t ($t > p$). The state of a new observation $\mathbf{y}_{t+1}$ (in this case $\mathbf{y}_{t+1}$ is the feature vector extracted from the curve acquired at the time step $t+1$) can be identified by assigning $\mathbf{y}_{t+1}$ to the state $\hat{z}_{t+1}$ using the MAP rule:

$$\hat{z}_{t+1} = \arg\max_{1\leq k\leq K} p(z_{t+1}=k|\mathbf{y}_1,\ldots,\mathbf{y}_{t+1};\hat{\boldsymbol{\Psi}}^{(t)}), \tag{5.39}$$

where

$$\begin{aligned} p(z_{t+1}=k|\mathbf{y}_1,\ldots,\mathbf{y}_{t+1};\hat{\boldsymbol{\Psi}}^{(t)}) &= p(z_{t+1}=k|\mathbf{y}_{t+1},\mathbf{r}_{t+1};\hat{\boldsymbol{\Psi}}^{(t)}) \\ &= \frac{\pi_k(\mathbf{r}_{t+1};\hat{\mathbf{w}}^{(t)})\mathcal{N}(\mathbf{y}_{t+1};\mathbf{B}_k^{T(t)}\mathbf{r}_{t+1},\hat{\boldsymbol{\Sigma}}_k^{(t)})}{\sum_{\ell=1}^K \pi_\ell(\mathbf{r}_{t+1};\hat{\mathbf{w}}^{(t)})\mathcal{N}(\mathbf{y}_{t+1};\mathbf{B}_\ell^{T(t)}\mathbf{r}_{t+1},\hat{\boldsymbol{\Sigma}}_\ell^{(t)})} \end{aligned} \tag{5.40}$$

is the posterior probability of the state (class) k for the new observation $\mathbf{y}_{t+1}$ given the previous observations $\mathbf{r}_{t+1}=(\mathbf{y}_t,\ldots,\mathbf{y}_{t-p+1})$ and the estimated models parameters up to time t, that is $\hat{\boldsymbol{\Psi}}^{(t)} = (\hat{\mathbf{w}}^{(t)},\hat{\boldsymbol{\Psi}}_1^{(t)},\ldots,\hat{\boldsymbol{\Psi}}_K^{(t)})$ with $\hat{\boldsymbol{\Psi}}_k^{(t)} = (\hat{\mathbf{B}}_k^{(t)},\hat{\boldsymbol{\Sigma}}_k^{(t)})$.

State prediction

The proposed ARHLP model can also be used to for state prediction for the future observation $\mathbf{y}_{t+1}$ ($t\geq p$), given the observed history up to time t. This consists of maximizing with respect to k the prediction probability

$$\begin{aligned} p(z_{t+1}=k|\mathbf{y}_1,\ldots,\mathbf{y}_t;\hat{\boldsymbol{\Psi}}^{(t)}) &= p(z_{t+1}=k|\mathbf{y}_t,\ldots,\mathbf{y}_{t-p+1};\hat{\boldsymbol{\Psi}}^{(t)}) \\ &= p(z_{t+1}=k|\mathbf{r}_{t+1};\hat{\boldsymbol{\Psi}}^{(t)}) \end{aligned} \tag{5.41}$$

which is none other than the probability of the logistic process given in Equation (5.3).

Observation prediction

On the other hand, the model can also be used to predict the future observation. The prediction of the future observed variable $\mathbf{y}_{t+1}$ is given by the conditional expectation over the distribution of $\mathbf{y}_{t+1}$, given the estimated model parameters $\hat{\boldsymbol{\Psi}}^{(t)}$:

$$\begin{aligned}
\hat{\mathbf{y}}_{t+1} &= \mathbb{E}[\mathbf{y}_{t+1}|\mathbf{y}_1,\ldots,\mathbf{y}_t;\hat{\boldsymbol{\Psi}}^{(t)}] \\
&= \int_{\mathbb{R}^d} \mathbf{y}_{t+1}\; p(\mathbf{y}_{t+1}|\mathbf{y}_1,\ldots,\mathbf{y}_t;\hat{\boldsymbol{\Psi}}^{(t)})d\mathbf{y}_{t+1} \\
&= \int_{\mathbb{R}^d} \mathbf{y}_{t+1}\; p(\mathbf{y}_{t+1}|\mathbf{r}_{t+1};\hat{\boldsymbol{\Psi}}^{(t)})d\mathbf{y}_{t+1} \\
&= \sum_{k=1}^{K} \pi_k(\mathbf{r}_{t+1};\hat{\mathbf{w}}^{(t)}) \int_{\mathbb{R}^d} \mathbf{y}_{t+1}\mathcal{N}\big(\mathbf{y}_{t+1};\hat{\mathbf{B}}_k^{T(t)}\mathbf{r}_{t+1},\hat{\boldsymbol{\Sigma}}_k^{(t)}\big)d\mathbf{y}_{t+1} \\
&= \sum_{k=1}^{K} \pi_k(\mathbf{r}_{t+1};\hat{\mathbf{w}}^{(t)})\hat{\mathbf{B}}_k^{T(t)}\mathbf{r}_{t+1} \qquad (5.42)
\end{aligned}$$

where $\mathbf{r}_{t+1} = (\mathbf{y}_t,\ldots,\mathbf{y}_{t-p+1})^T$.

The prediction can also be performed up to time $t+u$ ($u \geq 1$). In this case, we would have the predicted value

$$\hat{\mathbf{y}}_{t+u} = \sum_{k=1}^{K} \pi_k(\hat{\mathbf{r}}_{t+u};\hat{\mathbf{w}})\hat{\mathbf{B}}_k^T\hat{\mathbf{r}}_{t+u}$$

where $\hat{\mathbf{r}}_{t+u} = (\hat{\mathbf{y}}_{t+u-1},\ldots,\hat{\mathbf{y}}_{t+u-p})^T$. Then the state z_{t+u} would be predicted for $\hat{\mathbf{y}}_{t+u}$ by applying the MAP rule (5.39) for the corresponding predicted observation $\hat{\mathbf{r}}_{t+u}$, that is:

$$\hat{z}_{t+u} = \arg\max_{1\leq k\leq K} p(z_{t+u}=k|\hat{\mathbf{y}}_{t+u},\hat{\mathbf{r}}_{t+u};\hat{\boldsymbol{\Psi}}^{(t)}). \qquad (5.43)$$

5.3.3 Online formulation of the EM algorithm

In the previous section we presented the parameter estimation by the EM algorithm in the batch mode where the data are stored in advance. However, this is not usually the case. For example in a condition monitoring framework, the data could be acquired in real time so that one would have to decide about the operating state of the system in an online mode. This section is dedicated to the online learning of the ARHLP model where the parameters will be estimated recursively via a dedicated online EM algorithm.

Basing on Equations (5.32), (5.33) and (5.38), which represent the model parameters updates in a batch mode using the expected sufficient statistics (5.31) and (5.37), an online approach will be derived in this section. This online scheme, as seen in section 2.2.4 (Cappé and Moulines, 2009), consists of combining the old expected sufficient statistics and those estimated from the current observation, based on a step-size λ_t as follows:

$$\boldsymbol{S}_{.}^{(t)} = (1-\lambda_t)\boldsymbol{S}_{.}^{(t-1)} + \lambda_t\mathbb{E}[\boldsymbol{T}_{.}|\mathbf{y}_t,\boldsymbol{\Psi}^{(t-1)}] \qquad (5.44)$$

where the step-size λ_t satisfies $\sum_{t=1}^{\infty}\lambda_t = \infty$ and $\sum_{t=1}^{\infty}\lambda_t^2 < \infty$. According to (5.44), when λ_t is large, we tend to forget the past results and considers the estimation from the current observation. When λ_t is small, this implies that the past observations have a larger effect on the model parameters update. A possible choice of this value can be $\lambda_t = t^{-a}$ with $0.5 < a \leq 1$ (Cappé, 2009; Cappé and Moulines, 2009).

Thus, according to the incremental rule (5.44), the sufficient statistics given in (5.31) used for updating the parameters $(\mathbf{B}_k, \mathbf{\Sigma}_k)$, $(k = 1, \ldots, K)$ are updated recursively as follows:

$$\begin{aligned}
\boldsymbol{S}_k^{(t)} &= (1-\lambda_t)\boldsymbol{S}_k^{(t-1)} + \lambda_t \tau_{tk}^{(t-1)}, \\
\boldsymbol{S}_{rr,k}^{(t)} &= (1-\lambda_t)\boldsymbol{S}_{rr,k}^{(t-1)} + \lambda_t \tau_{tk}^{(t-1)} \mathbf{r}_t \mathbf{r}_t^T, \\
\boldsymbol{S}_{yr,k}^{(t)} &= (1-\lambda_t)\boldsymbol{S}_{yr,k}^{(t-1)} + \lambda_t \tau_{tk}^{(t-1)} \mathbf{r}_t \mathbf{y}_t^T, \\
\boldsymbol{S}_{yy,k}^{(t)} &= (1-\lambda_t)\boldsymbol{S}_{yy,k}^{(t-1)} + \lambda_t \tau_{tk}^{(t-1)} \mathbf{y}_t \mathbf{y}_t^T,
\end{aligned} \tag{5.45}$$

where the superscript t is used instead of the iteration q since in the online mode the iteration is denoted by the time step t.

As shown in Appendix A.11, the updating rule for AR parameters $\mathbf{B}_k$ $(k = 1, \ldots, K)$ yields in the following recursive formula:

$$\mathbf{B}_k^{(t+1)} = \mathbf{B}_k^{(t)} + \left[\boldsymbol{S}_{rr,k}^{(t)}\right]^{-1} \lambda_t \; \tau_{tk}^{(t)} \mathbf{r}_t (\mathbf{y}_t^T - \mathbf{r}_t^T \mathbf{B}_k^{(t)}). \tag{5.46}$$

The covariance matrix is recursively updated in a similar way as

$$\mathbf{\Sigma}_k^{(t+1)} = \frac{\boldsymbol{S}_{yy,k}^{(t)} - \mathbf{B}_k^{T(t+1)} \boldsymbol{S}_{yr,k}^{(t)}}{\boldsymbol{S}_k^{(t)}}. \tag{5.47}$$

Similarly, the parameter $\mathbf{w}$ of the logistic process is recursively updated based on a recursive update of the corresponding statistics given in (5.37), in the following way:

$$\begin{cases}
\text{Initialization:} & \\
\boldsymbol{S}_{rr,w}^{(t,1)} = (1-\lambda_t)\boldsymbol{S}_{rr,w}^{(t-1)} + \lambda_t \big(\mathbf{\Pi}_t(\mathbf{r}_t; \mathbf{w}^{(t-1)}) - \boldsymbol{\pi}(\mathbf{r}_t; \mathbf{w}^{(t-1)})\boldsymbol{\pi}^T(\mathbf{r}_t; \mathbf{w}^{(t-1)})\big) \otimes \mathbf{r}_t \mathbf{r}_t^T, & \\
\boldsymbol{S}_{wy,k}^{(t,1)} = (1-\lambda_t)\boldsymbol{S}_{wy,k}^{(t-1)} + \lambda_t \big(\boldsymbol{\tau}_t^{(t-1)} - \boldsymbol{\pi}(\mathbf{r}_t; \mathbf{w}^{(t-1)})\big) \otimes \mathbf{r}_t \, \cdot & (5.48a) \\
\text{Iterate: } \mathbf{w}^{(t,l+1)} = \mathbf{w}^{(t,l)} + \left[\boldsymbol{S}_{rr,w}^{(t,l)}\right]^{-1} \boldsymbol{S}_{wy,k}^{(t,l)} \text{ until convergence} & (5.48b) \\
\text{Set } \mathbf{w}^{(t+1)} = \mathbf{w}^{(t,l_{max})}. & (5.48c)
\end{cases}$$

where l denotes the iteration number and l_{max} denote the required number of iterations by the IRLS procedure (5.48b). At each time step, the IRLS procedure (5.48b) uses a single data point to update the parameter of the logistic model.

In the following section, the hidden sequence is modeled by a Markov process. The autoregressive relation is still assumed for the observation sequence, similarly as for the ARHLP model presented in the previous section. Therefore, the arising models are autoregressive HMMs (ARHMs). We first recall the ARHMM and then present the autoregressive non-homogeneous HMM approach.

5.4 Switching autoregressive non-homogeneous HMM

5.4.1 The general framework of the switching autoregressive HMM

This section briefly recalls the switching multivariate autoregressive Hidden Markov Model (Celeux et al., 2004; Frühwirth-Schnatter, 2006; Juang and Rabiner, 1985; Rabiner, 1989).

The observation sequence $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ is assumed to be generated by the following multivariate autoregressive model governed by a hidden Markov chain:

$$\mathbf{y}_t = \mathbf{B}_{z_t}\mathbf{y}_{t-1} + \mathbf{e}_t, \quad \mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_{z_t}) \quad \forall t = 2, \ldots, T, \tag{5.49}$$

where z_t $(t = 1, \ldots, n)$ is a hidden discrete-valued random variable in the finite set $\{1, \ldots, K\}$ representing the state of the system at time t, the $d \times d$ matrices $\mathbf{B}_{z_t}$ and $\mathbf{\Sigma}_{z_t}$ are respectively the matrix of autoregression parameters and the covariance matrix for the state z_t, and the $\mathbf{e}_t$'s are d dimensional independent random variables distributed according to a standard multivariate Gaussian distribution.

In this model, the sequence of the hidden states $\mathbf{z} = (z_1, \ldots, z_n)$ is assumed to be a K-states homogeneous Markov chain of first-order (Celeux et al., 2004; Frühwirth-Schnatter, 2006) parametrized by the initial state distribution $\pi = (\pi_1, \ldots, \pi_K)$ where $\pi_k = p(z_1 = k)$ is the probability of the state k $(k = 1, \ldots, K)$ and $\mathbf{A}$ the matrix of transition probabilities where $\mathbf{A}_{\ell k} = p(z_t = k|z_{t-1} = \ell)$ is the probability of transition from the state ℓ at time $t-1$ to the state k at time t with the constraint $\sum_{k=1}^{K} \mathbf{A}_{\ell k} = 1$ $\forall \ell = 1, \ldots, K$. Figure 5.3 gives a graphical representation for an ARHMM model of order 1. The parameter estimation for the ARHMM is performed by maximum

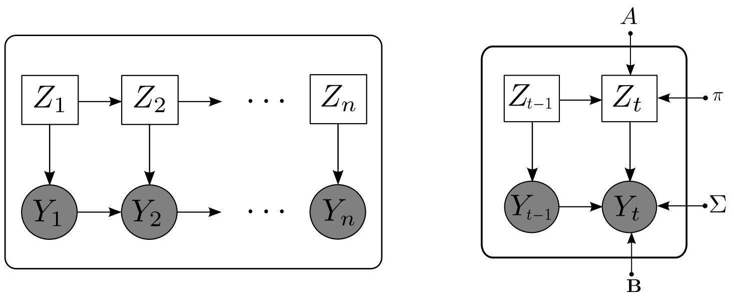


Figure 5.3: Graphical model structure for a multivariate autoregressive Hidden Markov Model of order 1 $(p = 1)$.

likelihood via the EM (Baum-Welch) algorithm (see for example (Celeux et al., 2004; Frühwirth-Schnatter, 2006)).

5.4.2 Non-homogeneous Markov chain

The approach recalled in the previous section concerns an homogeneous Markov chain. In this section we relax the stationarity assumption of the chain by considering a non-homogeneous Markov chain. More specifically, the time varying transition probabilities are modeled by a logistic distribution which depends on the previous observed variable $\mathbf{y}_{t-1}$ and a set of parameters $\mathbf{W}$. Thus, the transition probability from state ℓ to state k at each time t is given by:

$$\begin{aligned} \mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{W}) &= p(z_t = k|z_{t-1} = \ell, \mathbf{y}_{t-1};\mathbf{W}) \\ &= \frac{\exp\left(\boldsymbol{w}_k^{\ell\,T}\mathbf{y}_{t-1}\right)}{\sum_{k'=1}^{K}\exp\left(\boldsymbol{w}_{k'}^{\ell\,T}\mathbf{y}_{t-1}\right)}, \end{aligned} \tag{5.50}$$

where $\boldsymbol{w}_k^\ell$ is a d dimensional coefficients vector associated with the transition from state ℓ to state k, $\mathbf{w}_\ell = (\boldsymbol{w}_1^\ell, \ldots, \boldsymbol{w}_K^\ell)$ is the $(d \times K)$ matrix of parameters associated with the ℓth row of the state transition matrix $\mathbf{A}$ which characterizes all the transitions from state ℓ and $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_K)$ is the $(d \times K) \times K$ matrix of parameters associated with all the transitions. Note that since $\sum_{k=1}^{K}\mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{W}) = 1$ for all $\ell = 1, \ldots, K$, $\boldsymbol{w}_K^\ell$ is then set to the null vector to avoid identification problems.

Thus, the distribution of the latent sequence $\mathbf{z} = (z_1, \ldots, z_n)$, given the set of previous observations $(\mathbf{y}_1, \ldots, \mathbf{y}_{n-1})$ can be written as follows:

$$\begin{aligned} p(\mathbf{z}|\mathbf{y}_1,\ldots,\mathbf{y}_{n-1};\pi,\mathbf{W}) &= p(z_1;\pi)\prod_{t=2}^{n} p(z_t|z_{t-1},\mathbf{y}_{t-1};\mathbf{W}) \\ &= \prod_{k=1}^{K} p(z_1 = k;\pi)^{z_{1k}} \prod_{t=2}^{n}\prod_{k=1}^{K}\prod_{\ell=1}^{K} p(z_t = k|z_{t-1} = \ell, \mathbf{y}_{t-1};\mathbf{W})^{z_{t-1,\ell}z_{tk}} \\ &= \prod_{k=1}^{K}\pi_k^{z_{1k}} \prod_{t=2}^{n}\prod_{k=1}^{K}\prod_{\ell=1}^{K}\left(\frac{\exp\left(\boldsymbol{w}_k^{\ell\,T}\mathbf{y}_{t-1}\right)}{\sum_{h=1}^{K}\exp\left(\boldsymbol{w}_h^{\ell\,T}\mathbf{y}_{t-1}\right)}\right)^{z_{t-1,\ell}z_{tk}}. \end{aligned} \tag{5.51}$$

The resulting model is therefore a non-homogeneous ARHMM (AR-NH-HMM) which will be described in the next section.

5.4.3 Switching autoregressive non-homogeneous HMM

The model is defined by (5.49) for which the hidden state sequence is assumed to be a non-homogeneous stationary Markov chain, as defined in section 5.4.2, rather than a stationary Markov chain as for the standard ARHMM (Celeux et al., 2004; Douc et al., 2004; Frühwirth-Schnatter, 2006). The non-homogeneous HMM model has been initially proposed by Diebold et al. (1994) for a two-state HMM with continuous emission probabilities. Another non-homogeneous model is concerned with discrete variables (Sin and Kim, 1995). In this section we consider a multi-state autoregressive non-homogeneous HMM (AR-NH-HMM). This approach is linked to the Markov mixture of experts proposed by Meila and Jordan (1996) which is concerned with an univariate the batch mode learning The M-step includes a Gradient ascent procedure. In the model presented here, the parameter estimation using EM is given for the batch model as well as for the online mode. The M-step uses an IRLS procedure to estimate the parameters of the hidden process.

The data generation scheme with the AR-NH-HMM model

The data generation scheme with the AR-NH-HMM model, given a set of model parameters $\boldsymbol{\Psi} = (\pi, \mathbf{W}, \boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_K)$, is performed in a similar way as for the ARHLP model (c.f. section 5.3.1, except for the generation of the hidden sequence $\mathbf{z}$. In this case, the variable z_t for $t = 1, \ldots, n$ is generated as follows:

- Generate z_1 according to the multinomial distribution $\mathcal{M}(1, \pi_1, \ldots, \pi_K)$,
- Generate the hidden variable z_t $(t = 2, \ldots, n)$ according to the multinomial distribution $\mathcal{M}(1, \mathbf{a}_\ell(\mathbf{y}_{t-1}; \mathbf{w}_\ell))$ where $\mathbf{a}_\ell(\mathbf{y}_{t-1}; \mathbf{w}_\ell) = \big(\mathbf{A}_{\ell 1}(\mathbf{y}_{t-1}; \mathbf{w}_\ell), \ldots, \mathbf{A}_{\ell K}(\mathbf{y}_{t-1}; \mathbf{w}_\ell)\big)$ represents the K transition logistic probabilities from the state $z_{t-1} = \ell$ to the state $z_t = k$ $(k = 1, \ldots, K)$ at the time step t. In fact, these probabilities are those in the ℓth row of the time varying transition matrix. Therefore the corresponding parameter vector for the logistic function is $\mathbf{w}_\ell$.

The model is illustrated by the graphical representation shown in Figure 5.4.

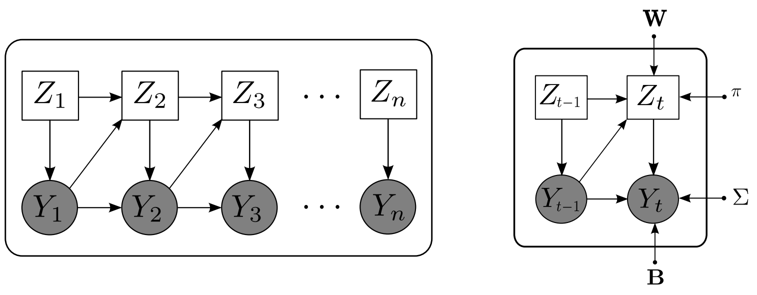


Figure 5.4: Graphical model structure for the multivariate autoregressive non-homogeneous HMM (AR-NH-HMM).

The next section is concerned with the unsupervised learning of the AR-NH-HMM model in a maximum likelihood framework using the EM algorithm.

5.4.4 The batch EM algorithm for parameter estimation

The unknown model parameter vector $\boldsymbol{\Psi} = (\pi, \mathbf{W}, \boldsymbol{\Psi}_1, \ldots, \boldsymbol{\Psi}_K)$ is estimated by the maximum likelihood method. The log-likelihood of $\boldsymbol{\Psi}$ for the observed sequence can be written as

$$\begin{aligned} \mathcal{L}(\boldsymbol{\Psi}; \mathbf{Y}) &= \log p(\mathbf{Y}; \boldsymbol{\Psi}) = \log \sum_{\mathbf{z}} p(\mathbf{Y}, \mathbf{z}; \boldsymbol{\Psi}) \\ &= \log \sum_{\mathbf{z}} p(\mathbf{z}|\mathbf{y}_1, \ldots, \mathbf{y}_{n-1}; \mathbf{W}) p(\mathbf{Y}|\mathbf{z}, \mathbf{y}_1, \ldots, \mathbf{y}_{n-1}; \boldsymbol{\Psi}). \end{aligned} \quad (5.52)$$

This log-likelihood can not be maximized in a closed-form. The EM algorithm (Baum et al., 1970; Dempster et al., 1977) will be used to perform the maximization.

In this framework, the complete-data log-likelihood is given as follows. By using the conditional independence assumptions for the model, that are:

(a) the observation $\mathbf{y}_t$ is conditionally independent of the other observations given the previous observation $\mathbf{y}_{t-1}$ and the current state z_t,

(b) the state z_t is conditionally independent of the other states given the previous one and the previous observation $\mathbf{y}_{t-1}$,

given an observation sequence $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ and a particular configuration of the latent process $\mathbf{z} = (z_1, \ldots, z_n)$ we therefore have:

$$\begin{aligned} p(\mathbf{Y}|\mathbf{z}, \mathbf{y}_1, \ldots, \mathbf{y}_{n-1}; \mathbf{\Psi}) &= p(\mathbf{y}_1|z_1; \mathbf{\Psi}_{z_1}) \prod_{t=2}^{n} p(\mathbf{y}_t|z_t, \mathbf{y}_{t-1}; \mathbf{\Psi}_{z_t}) \\ &\propto \prod_{t=2}^{n}\prod_{k=1}^{K} p(\mathbf{y}_t|z_t = k, \mathbf{y}_{t-1}; \mathbf{\Psi}_k)^{z_{tk}}. \end{aligned} \quad (5.53)$$

Then, by using the conditional distribution of the state sequence (c.f., Equation (5.51)), the likelihood of $\mathbf{\Psi}$ for the complete data $(\mathbf{Y}, \mathbf{z})$ can therefore be written as

$$\begin{aligned} p(\mathbf{Y}, \mathbf{z}; \mathbf{\Psi}) &= p(z_1; \pi) \prod_{t=2}^{n} p(z_t|z_{t-1}, \mathbf{y}_{t-1}; \mathbf{W}) \prod_{t=2}^{n} p(\mathbf{y}_t|z_t, \mathbf{y}_{t-1}; \mathbf{\Psi}_{z_t}) \\ &= \prod_{k=1}^{K} \pi_k^{z_{1k}} \prod_{t=2}^{n}\prod_{k=1}^{K}\prod_{\ell=1}^{K} \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{W})^{z_{t-1,\ell} z_{tk}} \prod_{t=2}^{n}\prod_{k=1}^{K} \mathcal{N}(\mathbf{y}_t; \mathbf{B}_k \mathbf{y}_{t-1}, \mathbf{\Sigma}_k)^{z_{tk}} \end{aligned} \quad (5.54)$$

and the complete-data log-likelihood is then obtained by taking the logarithm of (5.54), that is:

$$\begin{aligned} \mathcal{L}_c(\mathbf{\Psi}; \mathbf{Y}, \mathbf{z}) &= \sum_{k=1}^{K} z_{1k} \log \pi_k + \sum_{k=1}^{K}\sum_{\ell=1}^{K}\sum_{t=2}^{n} z_{t-1,\ell} z_{tk} \log \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{W}) \\ &\quad + \sum_{k=1}^{K}\sum_{t=2}^{n} z_{tk} \log \mathcal{N}(\mathbf{y}_t; \mathbf{B}_k \mathbf{y}_{t-1}, \mathbf{\Sigma}_k). \end{aligned} \quad (5.55)$$

Here we will consider the sufficient statistics of the model parameters in order to show the parallel between the batch approach and the online approach which will be presented later (c.f., section 5.4.5). From the first term of Equation (5.55), it can be seen that $\boldsymbol{T}_{1,k} = z_{1k}$ is the sufficient statistic for π_k. Similarly as in Equation (5.31), it can be deduced from the last term of Equation (5.55) that

$$\boldsymbol{T}_k = \sum_{t=2}^{n} z_{tk},\ \boldsymbol{T}_{rr,k} = \sum_{t=2}^{n} z_{tk} \mathbf{y}_{t-1} \mathbf{y}_{t-1}^T,\ \boldsymbol{T}_{yr,k} = \sum_{t=2}^{n} z_{tk} \mathbf{y}_{t-1} \mathbf{y}_t^T,\ \boldsymbol{T}_{yy,k} = \sum_{t=2}^{n} z_{tk} \mathbf{y}_t \mathbf{y}_t^T,$$

are the sufficient statistics for $\mathbf{B}_k$ and $\mathbf{\Sigma}_k$.

The optimization of the complete-data log-likelihood (5.55) with respect to the parameters $\mathbf{W}$ governing the time-varying transition matrix is a multinomial logistic regression problem which can be solved using the IRLS algorithm as

$$\mathbf{w}_\ell^{(l+1)} = \mathbf{w}_\ell^{(l)} + \left[\boldsymbol{T}_{rr,w\ell}^{(l)}\right]^{-1} \boldsymbol{T}_{r,w\ell}^{(l)}. \quad (5.56)$$

for $\ell = 1, \ldots, K$. In this updating formula, the statistics $\boldsymbol{T}^{(l)}_{rr,w\ell}$ and $\boldsymbol{T}^{(l)}_{r,w\ell}$ for the parameters $\mathbf{W}$ of the hidden process given the complete-data, are derived from (A.12) and are given by:

$$\begin{aligned}\boldsymbol{T}^{(l)}_{rr,w\ell} &= \frac{\partial^2 \mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{z})}{\partial \mathbf{w}_\ell \partial \mathbf{w}_\ell^T}\Big|^{-1}_{\mathbf{w}_\ell=\mathbf{w}_\ell^{(l)}} \\ &= -\sum_{t=2}^{n} z_{t-1,\ell}\left(\mathbf{A}^*_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell) - \mathbf{a}_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)})\mathbf{a}_{t\ell}^T(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)})\right) \otimes \mathbf{y}_{t-1}\mathbf{y}_{t-1}^T \quad (5.57)\end{aligned}$$

and

$$\boldsymbol{T}^{(l)}_{r,w\ell} = \frac{\partial \mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{z})}{\partial \mathbf{w}_\ell}\Big|_{\mathbf{w}_\ell=\mathbf{w}_\ell^{(l)}} = \sum_{t=2}^{n} z_{t-1,\ell}\big(\boldsymbol{z}_{tk} - \mathbf{a}_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)})\big) \otimes \mathbf{y}_{t-1} \quad (5.58)$$

where $\mathbf{a}_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell) = \big(\mathbf{A}_{\ell 1}(\mathbf{y}_{t-1};\mathbf{w}_\ell), \ldots, \mathbf{A}_{\ell,K-1}(\mathbf{y}_{t-1};\mathbf{w}_\ell)\big)^T$ in the vector of the $K-1$ logistic probabilities at the time step t associated with the ℓth row of the time varying transition matrix, $\mathbf{A}^*_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell) = \operatorname{diag}(\mathbf{a}_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell))$, and $\boldsymbol{z}_{tk} = (z_{t1}, \ldots, z_{t,K-1})^T$ is the $K-1$ dimensional vector of the binary indicator variables that correspond to the transition from the state ℓ to the state k ($k = 1, \ldots, K-1$). These transition probabilities correspond to the ℓth row of the transition matrix at each time step t.

The next section presents the parameter estimation in a batch (offline) mode from the incomplete-data using the EM algorithm.

The batch EM algorithm for the AR-NH-HMM

In the offline mode, the model parameters are estimated using a batch EM algorithm. The EM algorithm starts with an initial parameter $\boldsymbol{\Psi}^{(0)}$ and alternates between the E- and M-steps until convergence.

E-step: The E-step computes the conditional expected complete-data log-likelihood

$$\begin{aligned}Q(\boldsymbol{\Psi},\boldsymbol{\Psi}^{(q)}) &= \mathbb{E}\left[\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{z})|\mathbf{Y};\boldsymbol{\Psi}^{(q)}\right] \\ &= \sum_{k=1}^{K}\gamma_{1k}^{(q)}\log\pi_k + \sum_{k=1}^{K}\sum_{\ell=1}^{K}\sum_{t=2}^{n}\xi_{t\ell k}^{(q)}\log\mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{W}) \\ &\quad + \sum_{k=1}^{K}\sum_{t=2}^{n}\gamma_{tk}^{(q)}\log\mathcal{N}(\mathbf{y}_t;\mathbf{B}_k\mathbf{y}_{t-1},\boldsymbol{\Sigma}_k)\ , \quad (5.59)\end{aligned}$$

where

- $\gamma_{tk}^{(q)} = \mathbb{E}[z_{tk}|\mathbf{Y};\boldsymbol{\Psi}^{(q)}] = p(z_t = k|\mathbf{Y};\boldsymbol{\Psi}^{(q)})$ is the posterior probability of the state k ($k = 1, \ldots, K$) at time t ($t = 2, \ldots, T$) computed with the current parameter estimation $\boldsymbol{\Psi}^{(q)}$ given the whole observation sequence;

- $\xi_{t\ell k}^{(q)} = \mathbb{E}[z_{t-1,\ell} z_{tk} | \mathbf{Y}; \mathbf{\Psi}^{(q)}] = p(z_t = k, z_{t-1} = \ell | \mathbf{Y}; \mathbf{\Psi}^{(q)})$ is the joint posterior probability of the state k at time t and the state ℓ at time $t-1$ for $t = 2, \ldots, T$ and $\ell, k = 1, \ldots, K$ computed with the current parameter estimation $\mathbf{\Psi}^{(q)}$ given the whole observation sequence.

As shown in the expression of Q, this step requires the computation of the expected sufficient statistics through the probabilities $\gamma_{tk}^{(q)}$ and $\xi_{t\ell k}^{(q)}$. These probabilities are computed using the forward-backward procedure (Baum et al., 1970; Rabiner, 1989) from the probabilities α_{tk} and β_{tk} and are given by:

$$\gamma_{tk} = \frac{\alpha_{tk}\beta_{tk}}{\sum_{\ell=1}^{K} \alpha_{t\ell}\beta_{t\ell}}, \tag{5.60}$$

and

$$\xi_{t\ell k} = \frac{\alpha_{t-1,\ell}\mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{W}) p(\mathbf{y}_t | z_t = k; \mathbf{\Psi})\beta_{t,k}}{\sum_{\ell'=1}^{K}\sum_{k'=1}^{K} \alpha_{t-1,\ell'}\mathbf{A}_{\ell' k'}(\mathbf{y}_{t-1}; \mathbf{W}) p(\mathbf{y}_t | z_t = k'; \mathbf{\Psi})\beta_{tk'}}, \tag{5.61}$$

where the probabilities α_{tk} and β_{tk} are defined as

$$\alpha_{tk} = p(\mathbf{y}_1, \ldots, \mathbf{y}_t, z_t = k; \mathbf{\Psi}) , \tag{5.62}$$

which is the joint probability of observed sequence up to time t and the state k, and

$$\beta_{tk} = p(\mathbf{y}_{t+1}, \ldots, \mathbf{y}_n | z_t = k; \mathbf{\Psi}) , \tag{5.63}$$

is the probability of observing the rest of the sequence $(\mathbf{y}_{t+1}, \ldots, \mathbf{y}_n)$ knowing that the system starts from state k at time t. The probabilities α_{tk} and β_{tk} are computed recursively by the well-known Baum-Welch algorithm (Baum et al., 1970) (c.f., Appendix A.4) as follows. For $\ell, k = 1, \ldots, K$:

- $\alpha_{1k} = \pi_k p(\mathbf{y}_1 | z_1 = k; \mathbf{\Psi}_k)$ for $t = 1$,
- $\alpha_{tk} = \big[\sum_{\ell=1}^{K} \alpha_{t-1,\ell}\mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{W})\big] p(\mathbf{y}_t | z_t = k, \mathbf{y}_{t-1}; \mathbf{\Psi}_k)$ for $t = 2, \ldots, n$,

and the backward probabilities are computed in a similar as

- $\beta_{nk} = 1$,
- $\beta_{t\ell} = \sum_{k=1}^{K} \beta_{t+1,k}\mathbf{A}_{\ell k}(\mathbf{y}_t; \mathbf{W}) p(\mathbf{y}_{t+1} | z_{t+1} = k, \mathbf{y}_t; \mathbf{\Psi}_k)$ for $t = n-1, \ldots, 1$.

Thus, notice that the log-likelihood (5.52) can be computed from (A.4) by $\mathcal{L}(\mathbf{\Psi}; \mathbf{Y}) = \log \sum_{k=1}^{K} \alpha_{nk}$.

M-step: In this step, the value of the parameter $\mathbf{\Psi}$ is updated by computing the parameter $\mathbf{\Psi}^{(q+1)}$ which maximizes the Q-function with respect to $\mathbf{\Psi}$. In this case, the Q-function can be decomposed as

$$Q(\mathbf{\Psi}, \mathbf{\Psi}^{(q)}) = Q_{\pi}(\pi, \mathbf{\Psi}^{(q)}) + Q_{\mathbf{w}}(\mathbf{W}, \mathbf{\Psi}^{(q)}) + \sum_{k=1}^{K} Q_{\mathbf{\Psi}_k}(\mathbf{\Psi}_k, \mathbf{\Psi}^{(q)}), \tag{5.64}$$

where

$$Q_\pi(\pi, \mathbf{\Psi}^{(q)}) = \sum_{k=1}^{K} \gamma_{t1}^{(q)} \log \pi_k \ , \tag{5.65}$$

$$Q_{\mathbf{w}}(\mathbf{W}, \mathbf{\Psi}^{(q)}) = \sum_{\ell=1}^{K}\sum_{k=1}^{K}\sum_{t=2}^{n} \xi_{t\ell k}^{(q)} \log \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{W}) \ , \tag{5.66}$$

and

$$Q_{\mathbf{\Psi}_k}(\mathbf{\Psi}_k, \mathbf{\Psi}^{(q)}) = \sum_{t=2}^{n} \gamma_{tk}^{(q)} \log \mathcal{N}(\mathbf{y}_t; \mathbf{B}_k \mathbf{y}_{t-1}, \mathbf{\Sigma}_k). \tag{5.67}$$

The maximization of $Q(\mathbf{\Psi}, \mathbf{\Psi}^{(q)})$ with respect to $\mathbf{\Psi}$ is performed by separate maximizations of Q_π, $Q_{\mathbf{w}}$ and $Q_{\mathbf{\Psi}_k}$. The function Q_π is maximized w.r.t the non-negative initial state probabilities $(\pi_1, \ldots, \pi_K)$ subject to the constraint $\sum_{k=1}^{K} \pi_k = 1$ and provides the following update formula:

$$\pi_k^{(q+1)} = \gamma_{1k}^{(q)} \quad (k = 1, \ldots, K). \tag{5.68}$$

The maximizations of $Q_{\mathbf{\Psi}_k}$ with respect to $\mathbf{B}_k$ and $\mathbf{\Sigma}_k$ are performed in the same way as in the case of the maximizations of (5.25). The provided update formulas are then similar to (5.26) and (5.28) and are given by:

$$\mathbf{B}_k^{(q+1)} = \Big[\sum_{t=2}^{n} \gamma_{tk}^{(q)} \mathbf{y}_{t-1}\mathbf{y}_{t-1}^T\Big]^{-1} \sum_{t=2}^{n} \gamma_{tk}^{(q)} \mathbf{y}_{t-1}\mathbf{y}_t^T \tag{5.69}$$

and

$$\mathbf{\Sigma}_k^{(q+1)} = \frac{\sum_{t=2}^{n} \gamma_{tk}^{(q)} \mathbf{y}_t \mathbf{y}_t^T - \mathbf{B}_k^{(q+1)} \sum_{t=2}^{n} \gamma_{tk}^{(q)} \mathbf{y}_t \mathbf{y}_{t-1}^T}{\sum_{t=2}^{n} \gamma_{tk}^{(q)}}. \tag{5.70}$$

These updating formulas can be rewritten in function of the expected sufficient statistics as

$$\mathbf{B}_k^{(q+1)} = \Big[\boldsymbol{S}_{rr,k}^{(q)}\Big]^{-1} \boldsymbol{S}_{yr,k}^{(q)} \tag{5.71}$$

and

$$\mathbf{\Sigma}_k^{(q+1)} = \frac{\boldsymbol{S}_{yy,k}^{(q)} - \mathbf{B}_k^{T(q+1)} \boldsymbol{S}_{yr,k}^{(q)}}{\boldsymbol{S}_k^{(q)}} \tag{5.72}$$

where the expected sufficient statistics for the model parameters are given by:

$$\begin{aligned}
\boldsymbol{S}_k^{(q)} &= \mathbb{E}[\boldsymbol{T}_k|\mathbf{Y}, \boldsymbol{\Psi}^{(q)}] = \sum_{t=2}^{n} \tau_{tk}^{(q)}, \\
\boldsymbol{S}_{rr,k}^{(q)} &= \mathbb{E}[\boldsymbol{T}_{rr,k}|\mathbf{Y}, \boldsymbol{\Psi}^{(q)}] = \sum_{t=2}^{n} \tau_{tk}^{(q)} \mathbf{r}_t \mathbf{r}_t^T, \\
\boldsymbol{S}_{yr,k}^{(q)} &= \mathbb{E}[\boldsymbol{T}_{yr,k}|\mathbf{Y}, \boldsymbol{\Psi}^{(q)}] = \sum_{t=2}^{n} \tau_{tk}^{(q)} \mathbf{r}_t \mathbf{y}_t^T, \\
\boldsymbol{S}_{yy,k}^{(q)} &= \mathbb{E}[\boldsymbol{T}_{yy,k}|\mathbf{Y}, \boldsymbol{\Psi}^{(q)}] = \sum_{t=2}^{n} \tau_{tk}^{(q)} \mathbf{y}_t \mathbf{y}_t^T. \qquad (5.73)
\end{aligned}$$

The maximization of $Q_{\mathbf{w}}(\mathbf{W}, \boldsymbol{\Psi}^{(q)})$ is performed by maximizing $Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})$ with respect to $\mathbf{w}_\ell$ for $\ell = 1, \ldots, K$ where

$$\begin{aligned}
Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)}) &= \sum_{k=1}^{K} \sum_{t=2}^{n} \xi_{t\ell k}^{(q)} \log \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell) \\
&= \sum_{k=1}^{K} \sum_{t=2}^{n} \xi_{t\ell k}^{(q)} \log \frac{\exp(\boldsymbol{w}_k^{\ell\,T} \mathbf{y}_{t-1})}{\sum_{k=1}^{K} \exp(\boldsymbol{w}_k^{\ell\,T} \mathbf{y}_{t-1})}. \qquad (5.74)
\end{aligned}$$

This corresponds to solving K multinomial weighted logistic regression problems where the weights are the joint state posterior probabilities $\xi_{t\ell k}^{(q)}$. It is iteratively performed by the IRLS algorithm which provides the parameter update $\mathbf{w}_\ell^{(q+1)}$ $(\ell = 1, \ldots, K)$. In this case, each iteration of the IRLS algorithm is run as follows:

$$\mathbf{w}_\ell^{(l+1)} = \mathbf{w}_\ell^{(l)} - \Big[\frac{\partial^2 Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})}{\partial \mathbf{w}_\ell \partial \mathbf{w}_\ell^T}\Big]^{-1}_{\mathbf{w}_\ell = \mathbf{w}_\ell^{(l)}} \frac{\partial Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})}{\partial \mathbf{w}_\ell}\Big|_{\mathbf{w}_\ell = \mathbf{w}_\ell^{(l)}}. \qquad (5.75)$$

The calculation details of the Hessian matrix and the gradient of $Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})$ evaluated at $\mathbf{w}_\ell = \mathbf{w}_\ell^{(l)}$ are given in Appendix A.13. Each of the $K-1$ block matrices of the Hessian is a $d \times d$ matrix and is given by:

$$\frac{\partial^2 Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})}{\partial \boldsymbol{w}_k^\ell \partial \boldsymbol{w}_h^{\ell\,T}}\Big|_{\mathbf{w}_\ell = \mathbf{w}_\ell^{(l)}} = -\sum_{t=p+1}^{n} \tau_{t-1,\ell}^{(q)} \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell^{(l)}) \big(\delta_{kh} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1}; \mathbf{w}_h^{(l)})\big) \mathbf{y}_{t-1} \mathbf{y}_{t-1}^T (5.76)$$

The gradient vector $\frac{\partial Q_{\boldsymbol{w}}(\mathbf{w}, \boldsymbol{\Psi}^{(q)})}{\partial \mathbf{w}}$ is composed of $K-1$ gradient component vectors

$$\frac{\partial Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})}{\partial \mathbf{w}_\ell} = \Big(\frac{\partial Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})}{\partial \boldsymbol{w}_1^\ell}, \ldots, \frac{\partial Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})}{\partial \boldsymbol{w}_{K-1}^\ell}\Big)^T, \qquad (5.77)$$

where each gradient component vector k $(k = 1, \ldots, K-1)$ is in $\mathbb{R}^d$ and is given by:

$$\frac{\partial Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})}{\partial \boldsymbol{w}_k^\ell}\Big|_{\mathbf{w}_\ell = \mathbf{w}_\ell^{(l)}} = \sum_{t=2}^{n} \big(\xi_{t\ell k}^{(q)} - \tau_{t-1,\ell}^{(q)} \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell^{(l)})\big) \mathbf{y}_{t-1}. \qquad (5.78)$$

The Hessian and the gradient of $Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})$ evaluated at $\mathbf{w}_\ell = \mathbf{w}_\ell^{(l)}$ can also be expressed in a matrix form as in equations (5.35) and (5.36) respectively as follows:

$$\frac{\partial^2 Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})}{\partial \mathbf{w}_\ell \partial \mathbf{w}_\ell^T}\Big|_{\mathbf{w}_\ell=\mathbf{w}_\ell^{(l)}} = -\sum_{t=2}^{n} \tau_{t-1,\ell}^{(q)} \Big(\mathbf{A}_{t\ell}^*(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)}) - \mathbf{a}_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)})\mathbf{a}_{t\ell}^T(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)})\Big) \otimes \mathbf{y}_{t-1}\mathbf{y}_{t-1}^T \tag{5.79}$$

and

$$\frac{\partial Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \boldsymbol{\Psi}^{(q)})}{\partial \mathbf{w}_\ell}\Big|_{\mathbf{w}_\ell=\mathbf{w}_\ell^{(l)}} = \sum_{t=2}^{n} \big(\boldsymbol{\xi}_{t\ell}^{(q)} - \tau_{t-1,\ell}^{(q)}\mathbf{a}_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)})\big) \otimes \mathbf{y}_{t-1} \tag{5.80}$$

where $\boldsymbol{\xi}_{t\ell}^{(q)} = (\xi_{t\ell 1}^{(q)}, \ldots, \xi_{t\ell,K-1}^{(q)})^T$ is a $K-1$ dimensional vector, each of the $K-1$ elements represents the joint posterior probabilities at time t of state ℓ and state k ($k = 1, \ldots, K-1$), $\mathbf{a}_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell) = \big(\mathbf{A}_{\ell 1}(\mathbf{y}_{t-1};\mathbf{w}_\ell), \ldots, \mathbf{A}_{\ell,K-1}(\mathbf{y}_{t-1};\mathbf{w}_\ell)\big)^T$ are the $K-1$ logistic probabilities at the time step t associated with the ℓth row of the time varying transition matrix and $\mathbf{A}_{t\ell}^*(\mathbf{y}_{t-1};\mathbf{w}_\ell) = \mathrm{diag}(\mathbf{a}_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell))$.

By considering the following expected statistics

$$\begin{aligned}
\boldsymbol{S}_{rr,w\ell}^{(q,l)} &= \sum_{t=2}^{n} \tau_{t-1,\ell}^{(q)} \Big(\mathbf{A}_{t\ell}^*(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)}) - \mathbf{a}_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)})\mathbf{a}_{t\ell}^T(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)})\Big) \otimes \mathbf{y}_{t-1}\mathbf{y}_{t-1}^T \\
\boldsymbol{S}_{r,w\ell}^{(q,l)} &= \sum_{t=2}^{n} \big(\boldsymbol{\xi}_{t\ell}^{(q)} - \tau_{t-1,\ell}^{(q)}\mathbf{a}_{t\ell}(\mathbf{y}_{t-1};\mathbf{w}_\ell^{(l)})\big) \otimes \mathbf{y}_{t-1}
\end{aligned} \tag{5.81}$$

we can then see that the M-step update formula of $\mathbf{w}_\ell$ (5.34) takes the following form:

$$\mathbf{w}_\ell^{(l+1)} = \mathbf{w}_\ell^{(l)} + \Big[\boldsymbol{S}_{rr,w\ell}^{(q,l)}\Big]^{-1} \boldsymbol{S}_{r,w\ell}^{(q,l)}. \tag{5.82}$$

The pseudo code 8 summarizes the batch EM algorithm for the AR-NH-HMM model.

Algorithm 8 Pseudo code of the batch EM (Baum-Welch) algorithm for the proposed AR-NH-HMM model.

Inputs: An observation sequence $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ and the number of states K.

```
1: Initialize: $\mathbf{\Psi}^{(0)} = (\pi^{(0)}, \mathbf{W}^{(0)}, \mathbf{\Psi}_1^{(0)}, \ldots, \mathbf{\Psi}_K^{(0)})$ with $\mathbf{\Psi}_k^{(0)} = (\mathbf{B}_k^{(0)}, \mathbf{\Sigma}_K^{(0)})$.
2: fix a threshold $\epsilon > 0$
3: set $q \leftarrow 0$ (EM iteration)
4: while increment in log-likelihood $> \epsilon$ do
5:   E-Step (Forward-Backward procedure):
6:   for $k = 1, \ldots, K$ do
7:     compute $\tau_{tk}^{(q)}$ for $t = 2, \ldots, n$ using Equation (5.60)
8:     for $\ell = 1, \ldots, K$ do
9:       compute $\xi_{t\ell k}^{(q)}$ for $t = 2, \ldots, n$ using Equation (5.61)
10:     end for
11:   end for
12:   M-Step:
13:   compute $\pi_k^{(q+1)}$ using Equation (5.68)
14:   for $k = 1, \ldots, K$ do
15:     compute $\mathbf{B}_k^{(q+1)}$ using Equation (5.69)
16:     compute $\mathbf{\Sigma}_k^{(q+1)}$ using Equation (5.70)
17:     IRLS:
18:     Initialize: set $\mathbf{W}^{(l)} = \mathbf{W}^{(q)}$
19:     set a threshold $\delta > 0$
20:     for $\ell = 1, \ldots, K$ do
21:       $l \leftarrow 0$ (IRLS iteration)
22:       while increment in $Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \mathbf{\Psi}^{(q)}) > \delta$ do
23:         compute $\mathbf{w}_\ell^{(l+1)}$ using Equation (5.75)
24:         $l \leftarrow l + 1$
25:       end while
26:     end for
27:     $\mathbf{W}^{(q+1)} \leftarrow \mathbf{W}^{(l)}$
28:     $q \leftarrow q + 1$
29:   end for
30: end while
31: $\hat{\mathbf{\Psi}} = \mathbf{\Psi}^{(q)}$
```

Outputs: $\hat{\mathbf{\Psi}} = (\hat{\pi}, \hat{\mathbf{W}}, \hat{\mathbf{\Psi}}_1, \ldots, \hat{\mathbf{\Psi}}_K)$

State estimation

The use of the proposed AR-NH-HMM to make decision can be performed as follows. For example, suppose we are monitoring a dynamical system basing on acquired observations up to time t ($t > 1$) basing on the AR-NH-HMM model. The monitoring task can then be performed in two steps. The model parameters are first estimated in an unsupervised context from the given training observation sequence. Then, the state probability $p(z_{t+1} = k|\mathbf{y}_1, \ldots, \mathbf{y}_{t+1}; \hat{\boldsymbol{\Psi}}^{(t)})$, which can be seen as a posterior state probability given the parameters $\hat{\boldsymbol{\Psi}}^{(t)}$ estimated from the observation sequence up to time t, can be computed in a recursive manner as follows:

$$\begin{aligned}
p(z_{t+1} = k|\mathbf{y}_1, \ldots, \mathbf{y}_{t+1}; \hat{\boldsymbol{\Psi}}^{(t)}) &= \frac{p(z_{t+1} = k, \mathbf{y}_1, \ldots, \mathbf{y}_{t+1}; \hat{\boldsymbol{\Psi}}^{(t)})}{p(\mathbf{y}_1, \ldots, \mathbf{y}_{t+1}; \hat{\boldsymbol{\Psi}}^{(t)})} \\
&= \frac{\alpha_{t+1,k}(\hat{\boldsymbol{\Psi}}^{(t)})}{\sum_{h=1}^{K} \alpha_{t+1,h}(\hat{\boldsymbol{\Psi}}^{(t)})} \\
&= \frac{[\sum_{\ell=1}^{K} \alpha_{t\ell}(\hat{\boldsymbol{\Psi}}^{(t)}) A_{\ell k}(\mathbf{y}_t; \hat{\mathbf{W}}^{(t)})] p(\mathbf{y}_{t+1}|z_{t+1} = k; \hat{\boldsymbol{\Psi}}^{(t)}))}{\sum_{h=1}^{K}[\sum_{\ell=1}^{K} \alpha_{t\ell}(\hat{\boldsymbol{\Psi}}^{(t)}) A_{\ell h}(\mathbf{y}_t; \hat{\mathbf{W}}^{(t)})] p(\mathbf{y}_{t+1}|z_{t+1} = h; \hat{\boldsymbol{\Psi}}^{(t)})}
\end{aligned} \tag{5.83}$$

where $\alpha_{tk}(\hat{\boldsymbol{\Psi}}^{(t)})$ are the forward probabilities computed up to time t. The decision about the state of a new acquired observation $\mathbf{y}_{t+1}$ can then be made by computing the optimal state sequence through the Viterbi algorithm Viterbi (1967) (see Appendix A.6).

State prediction

The model can also be used to make state prediction for the future observation $\mathbf{y}_{t+1}$ ($t \geq p$) given the observations up to time t. This consists of computing the prediction (or prior) probability $p(z_{t+1} = k|\mathbf{y}_1, \ldots, \mathbf{y}_t; \hat{\boldsymbol{\Psi}}^{(t)})$ which is computed as follows:

$$\begin{aligned}
p(z_{t+1} = k|\mathbf{y}_1, \ldots, \mathbf{y}_t; \hat{\boldsymbol{\Psi}}^{(t)}) &= \frac{p(z_{t+1} = k, \mathbf{y}_1, \ldots, \mathbf{y}_t; \hat{\boldsymbol{\Psi}}^{(t)})}{p(\mathbf{y}_1, \ldots, \mathbf{y}_t; \hat{\boldsymbol{\Psi}}^{(t)})} \\
&= \frac{\sum_{\ell=1}^{K} p(z_{t+1} = k, z_t = \ell, \mathbf{y}_1, \ldots, \mathbf{y}_t; \hat{\boldsymbol{\Psi}}^{(t)})}{\sum_{h=1}^{K} p(z_t = h, \mathbf{y}_1, \ldots, \mathbf{y}_t; \hat{\boldsymbol{\Psi}}^{(t)})} \\
&= \frac{\sum_{\ell=1}^{K} p(z_{t+1} = k|z_t = \ell, \mathbf{y}_t; \hat{\mathbf{W}}^{(t)}) p(z_t = \ell, \mathbf{y}_1, \ldots, \mathbf{y}_t; \hat{\boldsymbol{\Psi}}^{(t)})}{\sum_{h=1}^{K} \alpha_{th}(\hat{\boldsymbol{\Psi}}^{(t)})} \\
&= \frac{\sum_{\ell=1}^{K} \mathbf{A}_{\ell k}(\mathbf{y}_t; \hat{\mathbf{W}}^{(t)}) \alpha_{t\ell}(\hat{\boldsymbol{\Psi}}^{(t)})}{\sum_{h=1}^{K} \alpha_{th}(\hat{\boldsymbol{\Psi}}^{(t)})}
\end{aligned} \tag{5.84}$$

where in the third step we used the fact that conditionally on z_t, z_{t+1} depends only on the previous observation $\mathbf{y}_t$. This show that prediction is equivalent to propagating the forward probabilities without incorporating the missing observations $\mathbf{y}_{t+1}$.

Observation prediction

On the other hand, in a context of time series prediction, one can also use the probabilities given by (5.84) (which can be seen as the prior state probabilities) to make prediction on the future observation $\mathbf{y}_{t+1}$, that is:

$$\hat{\mathbf{y}}_{t+1} = \sum_{k=1}^{K} p(z_{t+1} = k|\mathbf{y}_1, \ldots, \mathbf{y}_t; \hat{\boldsymbol{\Psi}}^{(t)})\hat{\mathbf{B}}_k^{(t)T}\mathbf{y}_t. \tag{5.85}$$

In the next section we will formulate an online algorithm in which the parameters are recursively updated as the learning proceeds.

5.4.5 Online version of the EM algorithm

This section regards to the online framework in which the data arrive one at a time. We therefore formulate an online learning of the switching autoregressive non-homogeneous Hidden Markov Model (AR-NH-HMM). The online EM algorithms (Samé et al., 2007; Sato and Ishii, 2000; Titterington, 1984) provide an adapted framework to train probabilistic latent data models. In particular, for HMMs, for which earlier online approaches have been introduced by Baldi and Chauvin (1994), one can distinguish the recent contributions for continuous HMMs (Cappé, 2009) or discrete HMMs (Mongillo and Deneve, 2008). In this online mode, the model parameters can be recursively updated by updating the expected sufficient statistics of the model parameters (Cappé, 2009; Cappé and Moulines, 2009; Ng et al., 2006) at the E-step, while keeping the M-step free (c.f., section 2.2.4). In this section we adopt this strategy of the online EM algorithm to train the AR-NH-HMM. Let us recall that, as seen in section 2.2.4 (Cappé, 2009; Cappé and Moulines, 2009), the recursive update strategy for the expected sufficient statistics consists of combining the old expected sufficient statistics and those estimated from the current observation by using a decreasing step-size λ_t as follows:

$$\boldsymbol{S}_{\cdot}^{(t)} = (1-\lambda_t)\boldsymbol{S}_{\cdot}^{(t-1)} + \lambda_t\mathbb{E}[\boldsymbol{T}_{\cdot}|\mathbf{y}_t, \boldsymbol{\Psi}^{(t-1)}]. \tag{5.86}$$

Thus, the updating formulas for the expected sufficient statistics for the AR-NH-HMM model are derived as follows:

$$\begin{aligned}
\boldsymbol{S}_k^{(t)} &= (1-\lambda_t)\boldsymbol{S}_k^{(t-1)} + \lambda_t\gamma_{tk}^{(t-1)}\\
\boldsymbol{S}_{\ell k}^{(t)} &= (1-\lambda_t)\boldsymbol{S}_{\ell k}^{(t-1)} + \lambda_t\xi_{t\ell k}^{(t-1)}\\
\boldsymbol{S}_{yy,k}^{(t)} &= (1-\lambda_t)\boldsymbol{S}_{yy,k}^{(t-1)} + \lambda_t\gamma_{tk}^{(t-1)}\mathbf{y}_t\mathbf{y}_t^T\\
\boldsymbol{S}_{yr,k}^{(t)} &= (1-\lambda_t)\boldsymbol{S}_{yr,k}^{(t-1)} + \lambda_t\gamma_{tk}^{(t-1)}\mathbf{y}_{t-1}\mathbf{y}_t^T\\
\boldsymbol{S}_{rr,k}^{(t)} &= (1-\lambda_t)\boldsymbol{S}_{rr,k}^{(t-1)} + \lambda_t\gamma_{tk}^{(t-1)}\mathbf{y}_{t-1}\mathbf{y}_{t-1}^T.
\end{aligned} \tag{5.87}$$

Based on these updated expected sufficient statistics, the online updating rules for the AR model parameters $\mathbf{B}_k$ and $\boldsymbol{\Sigma}_k$ $(k = 1, \ldots, K)$ are given by:

$$\begin{aligned}
\mathbf{B}_k^{(t+1)} &= \left[\boldsymbol{S}_{rr,k}^{(t)}\right]^{-1}\boldsymbol{S}_{yr,k}^{(t)}\\
\boldsymbol{\Sigma}_k^{(t+1)} &= \frac{\boldsymbol{S}_{yy,k}^{(t)} - \mathbf{B}_k^{(t+1)}\boldsymbol{S}_{yr,k}^{(t)}}{\boldsymbol{S}_k^{(t)}}
\end{aligned} \tag{5.88}$$

and the initial state distribution is updated as

$$\pi_k^{(t+1)} = \boldsymbol{S}_k^{(1)} \quad (k = 1, \ldots, K). \tag{5.89}$$

The parameters $\mathbf{W}$ associated with the time varying transition probabilities are updated based on the statistics $\boldsymbol{S}_{rr,w\ell}$ and $\boldsymbol{S}_{r,w\ell}$ as follows:
For $\ell = 1, \ldots, K$:

$$\begin{cases} \text{Initialization:} \\ \boldsymbol{S}_{rr,w\ell}^{(t,1)} = (1-\lambda_t)\boldsymbol{S}_{rr,w\ell}^{(t-1)} \\ \qquad +\lambda_t \tau_{t-1,\ell}^{(t-1)} \left(\mathbf{A}_{t\ell}^*(\mathbf{y}_{t-1}; \mathbf{w}_\ell^{(t-1)}) - \mathbf{a}_{t\ell}(\mathbf{y}_{t-1}; \mathbf{w}_\ell^{(t-1)})\mathbf{a}_{t\ell}^T(\mathbf{y}_{t-1}; \mathbf{w}_\ell^{(t-1)})\right) \otimes \mathbf{y}_{t-1}\mathbf{y}_{t-1}^T \\ \boldsymbol{S}_{r,w\ell}^{(t,1)} = (1-\lambda_t)\boldsymbol{S}_{r,w\ell}^{(t-1)} + \lambda_t\big(\boldsymbol{\xi}_{t\ell}^{(t-1)} - \tau_{t-1,\ell}^{(t-1)}\mathbf{a}_{t\ell}(\mathbf{y}_{t-1}; \mathbf{w}_\ell^{(t-1)})\big) \otimes \mathbf{y}_{t-1}, & (5.90a) \\ \text{Iterate: } \mathbf{w}_\ell^{(t,l+1)} = \mathbf{w}_\ell^{(t,l)} + \left[\boldsymbol{S}_{rr,w\ell}^{(t,l)}\right]^{-1} \boldsymbol{S}_{r,w\ell}^{(t,l)} \text{ until convergence,} & (5.90b) \\ \text{Set } \mathbf{w}_\ell^{(t+1)} = \mathbf{w}_\ell^{(t,l_{max})}, & (5.90c) \end{cases}$$

where l denotes the IRLS iteration number, and l_{max} denote the number of iterations at convergence of the IRLS procedure (5.90b).

5.5 Experimental study

This section describes the results obtained with the ARHLP and the ARHMM for experiments conducted on a multidimensional synthetic data set. We perform comparisons with the standard HMM and ARHMM models. The questions these experiments aim at addressing are the ability to reconstruct the true underlying dynamic process.

5.5.1 Initialization and stopping rule

The EM algorithms for the ARHLP and the ARHMM are initialized as follows:

- the internal IRLS algorithm is initialized with random parameters $\mathbf{W}$ only for the first EM iteration ($q = 0$). For the rest of EM iterations, the IRLS algorithm is initialized with $\mathbf{w}^{(q)}$ obtained at the previous EM iteration.
- to initialize $\mathbf{B}_k$, we segment randomly the multivariate sequence into K regimes and on each regime k we fit an autoregressive model. We then estimate an initial value of the covariance matrix $\boldsymbol{\Sigma}_k$ for each model.

Several EM tries are performed and the solution providing the highest log-likelihood is chosen. For the first EM try we initialize $\mathbf{B}_k$ randomly and we set $\boldsymbol{\Sigma}_k = \mathbf{I}_d$ ($k = 1, \ldots, K$).

The batch EM algorithms are stopped when the relative variation of the optimized log-likelihood functions between two iterations is below a predefined threshold ($|\frac{\mathcal{L}^{(q+1)} - \mathcal{L}^{(q)}}{\mathcal{L}^{(q)}}| \leq 10^{-6}$ for EM and Baum-Welch, and $|\frac{Q_{\boldsymbol{w}}(\mathbf{w}^{(q+1)}) - Q_{\boldsymbol{w}}(\mathbf{w}^{(q)})}{Q_{\boldsymbol{w}}(\mathbf{w}^{(q)})}| \leq 10^{-6}$ for the IRLS) or when the iteration number reached a maximal number (1000 for both EM and Baum-Welch and 50 for IRLS). For the online mode, the learning is performed for each data item from the observation sequence until all the observations are processed.

5.5.2 Simulated multidimensional sequence

We considered a simulated observation sequence in $\mathbb{R}^2$ which consists with of $K = 2$ autoregressive models of order $p = 1$ over $n = 500$ time steps. The parameters $(\mathbf{w}, \mathbf{B}_1, \ldots, \mathbf{B}_K, \mathbf{\Sigma}_1, \ldots, \mathbf{\Sigma}_K)$ used for simulations are:

$$\mathbf{B}_1 = \left(\begin{array}{cc} 0.95 & 0.3 \\ 0 & 0.95 \end{array} \right), \; \mathbf{B}_2 = \left(\begin{array}{cc} 0.9 & 0 \\ -0.5 & 0.9 \end{array} \right) \; \boldsymbol{w}_1 = [35, 7]^T, \mathbf{\Sigma}_1 = \mathbf{\Sigma}_2 = \mathbf{I}_2.$$

The data are simulated according to the ARHLP model. Figure 5.5 shows an example of a simulated observation sequence and the corresponding time-varying logistic probabilities.

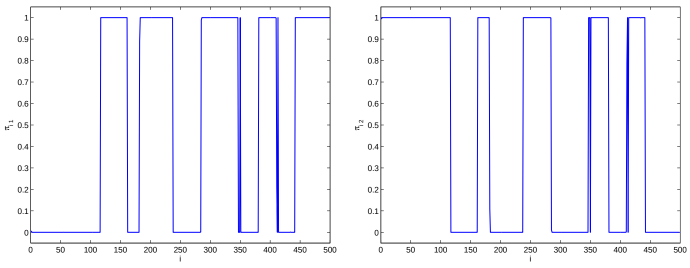


Figure 5.5: The logistic proportions for the simulations.

Figure 5.6 shows the simulated observation sequence and the corresponding plot in $\mathbb{R}^2$ where the data are labeled according to the true sequence.

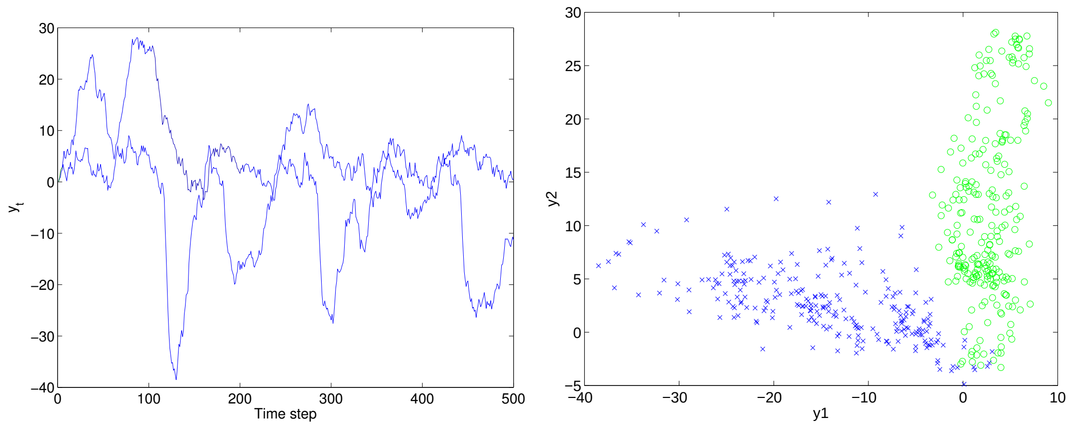


Figure 5.6: Example of simulated 2-dimentional observation sequence represented in function of the time (left) and in the plan (right).

5.5.3 Obtained results

Figure 5.7 shows the logistic probabilities and the posterior probabilities over time, obtained with the ARHLP model. The right plot of this figure shows the sequence labeled according to the estimated MAP sequence.

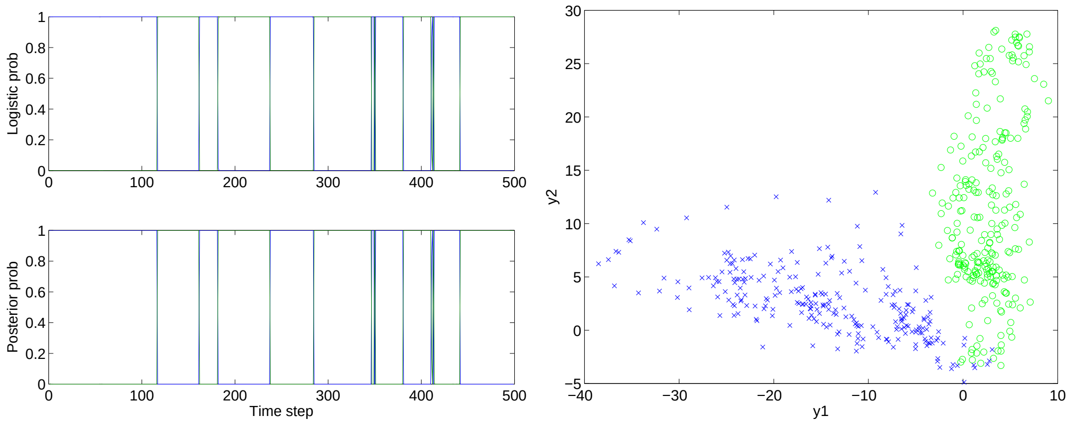


Figure 5.7: Results obtained with the ARHLP model.

Figure 5.8 shows the probabilities associated with the ARHMM model and the corresponding estimated labeled sequence. Let us recall that the prediction, filtering and smoothed provabilities are computed respectively according to Equation (5.84), (5.83) and (5.60).

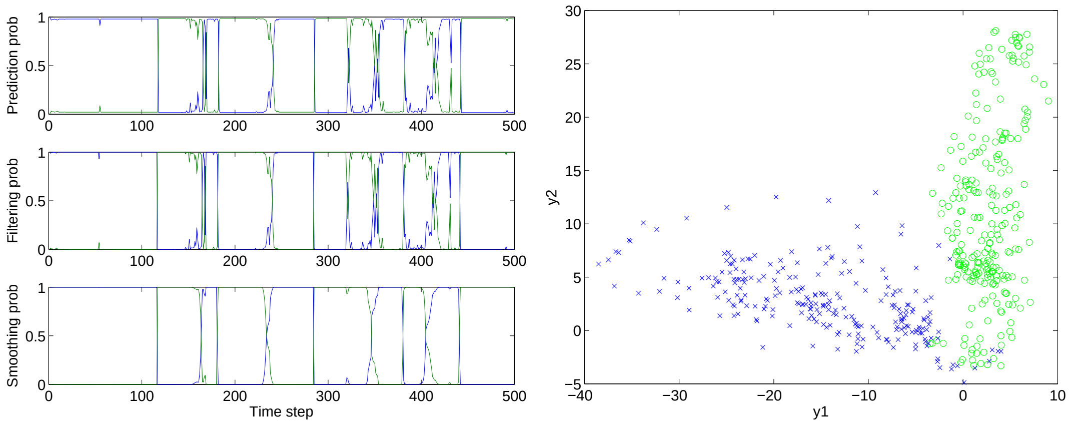


Figure 5.8: Results obtained with the AR-hom-HMM.

Finally, Figure 5.9 shows the results obtained with the AR-NH-HMM model. It can be seen that, while the data are generated according to the ARHLP model, the results obtained with the AR-NH-HMM, as it generalizes the ARHLP model, provides quasi similar results. On the other hand, it can be seen that, the AR-NH-HMM provide more accurate results compared to the homogeneous HMM. In particular, the difference can be observed on the prediction, filtering and smoothing probabilities.

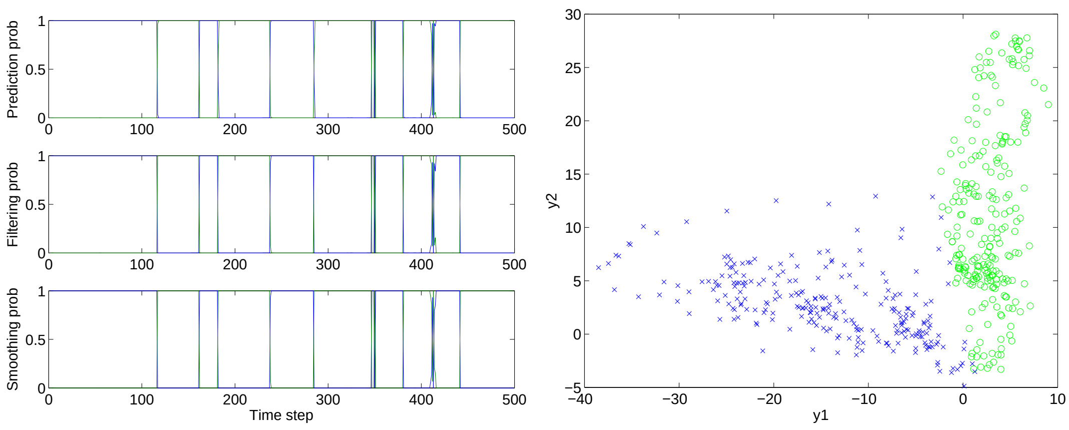


Figure 5.9: Results obtained with the AR-NH-HMM.

5.5.4 Simulated curve sequence

In this section we report results obtained with the proposed approaches on a curve sequence.

The curve sequence is composed of $n = 100$ curves simulated in a sequential manner as follows. We considered two states characterized by parameter vectors $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$. Each of the two states is composed of 30 curves. The rest of the curves are simulated in a sequential manner from state 1 to state 2 as follows:

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_1 + \lambda_t(\boldsymbol{\theta}_2 - \boldsymbol{\theta}_1)$$

where $\{\lambda_t\}$ is an increasing sequence of positive real numbers, regularly spaced, from 0.3 to 0.7. Each curve is composed of $m = 100$ points and is generated according to the RHLP model with three constant polynomial regimes ($K = 3, p = 0$). The used simulation parameters are given in Table 5.1. Figure 5.10 shows the simulated curves with the mean curves corresponding to the first and the final state.

	$\boldsymbol{\beta}_1$	$\boldsymbol{\beta}_2$	$\boldsymbol{\beta}_3$	$\boldsymbol{w}_1$	$\boldsymbol{w}_2$	$\boldsymbol{w}_3$	σ_1	σ_2	σ_3
$\boldsymbol{\theta}_1$	0	2.5	0	[49.1664,-1.5982]	[27.6774,-0.4567]	[0,0]	0.5	0.5	0.5
$\boldsymbol{\theta}_2$	0.1	5.4	0.1	[49.3626, -1.1978]	[27.7916, -0.4227]	[0,0]	0.55	0.55	0.55

Table 5.1: Simulation parameters for the curve sequence.

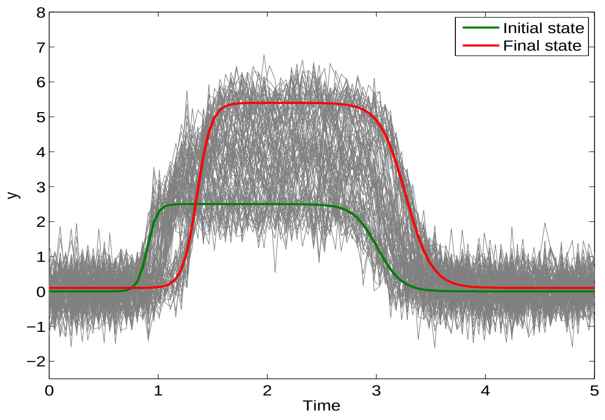


Figure 5.10: Simulated curve sequence: from state 1 to state 2.

5.5.5 Obtained results

Figures 5.11 and 5.13 show the probabilities associated with the ARHMM and AR-NH-HMM models (that is the prediction, filtering and smoothing probabilities). Figure 5.12 shows the logistic probabilities and the posterior probabilities over time, obtained with the ARHLP model.

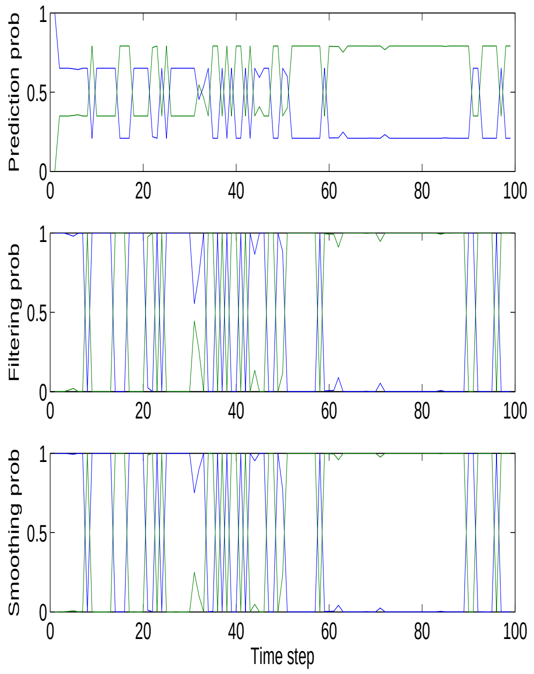


Figure 5.11: Results obtained with the AR-hom-HMM.

The obtained results clearly show that both the ARHLP and the AR-NH-HMM models are more in accordance with the true simulated states of the curves. At the

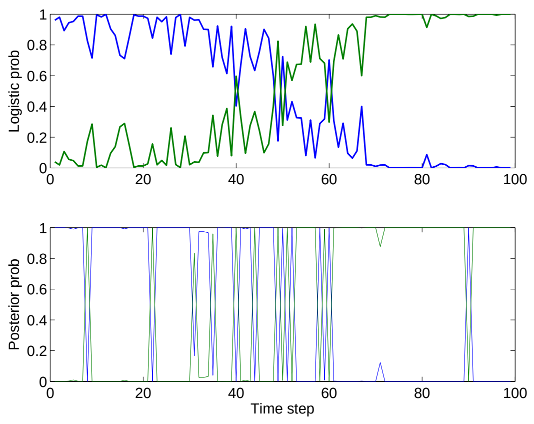


Figure 5.12: Results obtained with the ARHLP model.

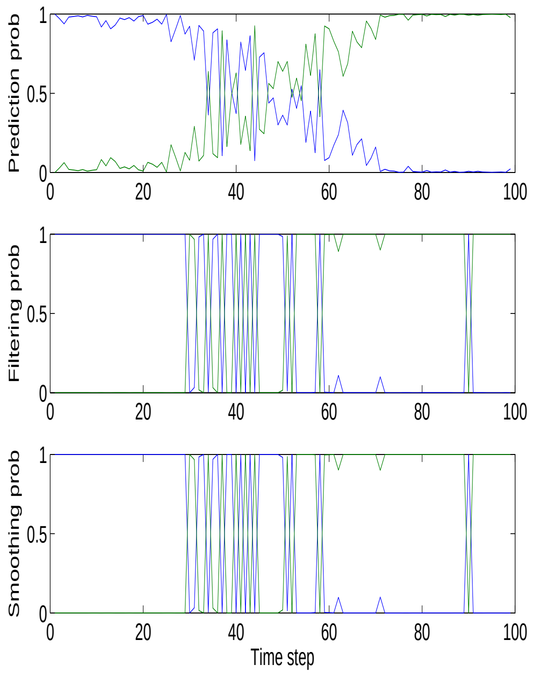


Figure 5.13: Results obtained with the AR-NH-HMM.

same time, it can be clearly observed that, the AR-NH-HMM provide more accurate results compared to the ARHLP model.

5.6 Summary

In this chapter, we developed two approaches for modeling a curve sequence using a two-stage strategy. The first stage consists in extracting features from the curves. We then studied two different approaches to the problem. The first is an autoregressive model governed by a hidden logistic process, and the second is an autoregressive model governed by a non-homogeneous HMM. We introduced two EM algorithms to train these dynamical probabilistic models.

In the formulations of the EM algorithms, the expected sufficient statistics are highlighted in order to derive online learning procedures.

An experimental study was conducted on multidimensional data, which are assumed to be extracted from curves. A second experimental study was performed on simulated curve sequences. The obtained results show the effectiveness of the proposed approaches.

In the next chapter, we apply the approaches developed in the previous chapters to monitoring railway switches.

Chapter 6

Application to the diagnosis of the railway switch mechanism

Contents

6.1 Introduction

The remote monitoring of the railway infrastructure components is of great interest for railway companies, including the French railway company (SNCF) and the managers of its infrastructure (RFF in France). A key problem consists of accurately detecting the presence of defects to alert the concerned maintenance service provider before faults occur. In this application context, pattern recognition approaches have been introduced in the LTN-INRETS Laboratory and have shown their good performances (Côme, 2009; Debiolles, 2007; Oukhellou, 1997). Precisely regarding switch mechanism diagnosis, another interesting approach has been proposed based on a continuous state-space model that consists of a Kalman filtering (Pedregal et al., 2004). However, in our study, the switch operating states are assumed to be finite.

In this chapter, we apply the methodologies developed in 3, 4 and 5 to the problem of railway switches diagnosis based on measured switch operation curves. First, we describe the railway monitoring system, the switch mechanism and the data. Then, we highlight the methodology used to perform the diagnosis task. The results provided by the different approaches are then introduced in both static and dynamical frameworks. Notice that hereafter, the terms "state" and "class" both indicate the switch operating state.

6.2 The switch mechanism

6.2.1 Definition of the switch mechanism

The switch mechanism is the railway infrastructure component that enables (high speed) trains to be guided from one track to another at a railway junction. Its proper functioning is required for the full availability of the transportation system. It is a vital organ whose operating state directly impacts the overall safety of the railway system; its monitoring is a key point of the actions of the maintenance teams. Figure 6.1 illustrates the composition of a switch mechanism. There are two movable switch points, which by their tilting, allow for guiding trains on the direct track or the deviated track. The tilting of the switch is generally operated through a linkage system controlled by an electrical motor.

Several failure modes (not exclusive) are observable: electrical problems on motors, mechanical problems on moving parts, civil engineering problems on the installation of the switch, etc.

6.3 The railway monitoring system

The railway monitoring system has three main parts:

- a sensors part for data acquisition by local sensors. The measured physical quantities may reflect the operating states of the railway infrastructure components including the switch mechanism.

  The measurement acquisition can be done by sensors on-board the track or portable memory oscilloscope. Obviously, in the latter case, the presence of a maintenance agent on the site is required. Preferably, the data acquisition could

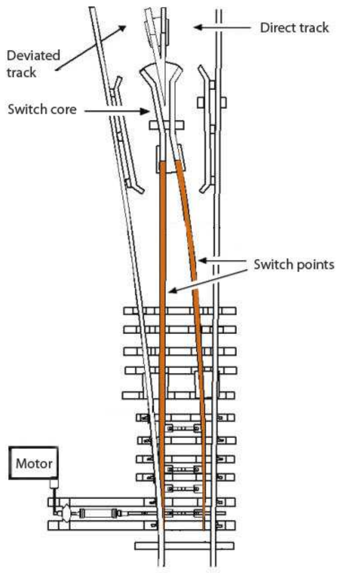


Figure 6.1: Switch mechanism.

be done by a remote monitoring system. Information are therefore available remotely inside the regulation office to make decision remotely and in real-time.

- a network part to enable for transmitting the measured data through an Ethernet-TCP/IP network protocol.
- a third component for the control/command, that is the management of alarms, response to incidents, etc.

Figure 6.2 shows the railway remote monitoring system used by SNCF with its three main layers.

6.3.1 The switch operation

A switch operation consists of five successive mechanical movements of the various components associated with the switch points. These movements are reflected on the shape of the curve through different operating phases:

- starting phase,
- points unlocking,
- points translation,
- points locking,
- friction phase.

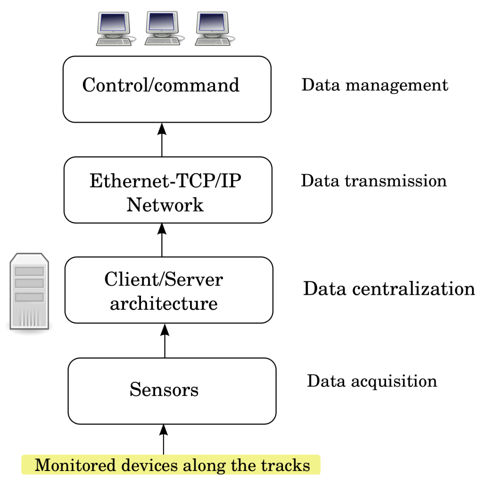


Figure 6.2: The railway remote monitoring system layout.

Starting phase

This phase corresponds to the period between the activation of the motor and the starting of the switch operation itself. The duration of this phase and the power consumption depend on the length and the type of the cable that connects the motor and the movable points.

Points unlocking

At the end of the initial phase, the motor moves the unlocking organ in order to prepare the mechanical translation. In this phase, the motor unlocks the switch points and makes them open up a position allowing for the translation.

In the first half of this phase, the engine brake to prevent stalling. In the second, After that, the motor unlocks the points and makes them open up a position allowing for the translation.

The consumed power in this phase corresponds to the effort required by the motor for the rotation of the clamp lock system (the VCC system). An over-consumption of power at this stage of the operation may, according to its shape, corresponds to a lack of lubrication, a clamping problem,...

Points translation

This phase corresponds to the translation of the points. Their new positions allow for a change of direction of the trains along the tracks. Different kinds of failures can occur during this phase as the lack of lubrication, a misfit trade-off of the linkage in the case of multiple attack (the effort from the actuator to the points is transmitted through several mechanical fasterners), the presence of obstacle (stone, frost and ice,...)

Points locking

The points locking phase corresponds to the effort needed for locking the points. As in the unlocking phase, over-effort in the locking phase may reflect a lack of lubrication, a clamping problem,...

Friction phase

The end of the switch operation is indicated by the friction phase. In this phase, an additional effort is applied to ensure the locking and a limiting device (a mechanical friction system) is used to protect the motor of the switch mechanism and limits the consumed power. Changes in the power curve can occur if the limiting device is inoperative.

6.3.2 The data acquired during a switch operation

For our diagnosis problem, the exploited data are the curves of the electrical power consumed during the switch operations (see Figure 6.3). Each curve is composed of 564 points with a sampling frequency of 100 H_z (100 points per second).

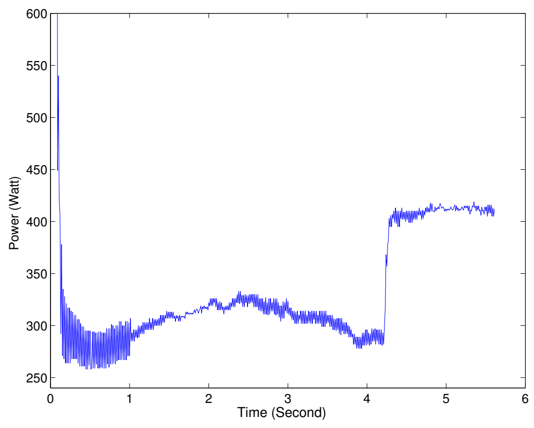


Figure 6.3: A curve showing the electrical power consumed during a switch operation.

The different phases involved in a switch operation can be illustrated in Figure 6.4.

The total duration of the switch operation is uniform for all curves, whereas the durations of each phase are unknown and may vary according to the operating state of the switch mechanism.

The described movements involved during a switch operation are represented in Figure 6.5.

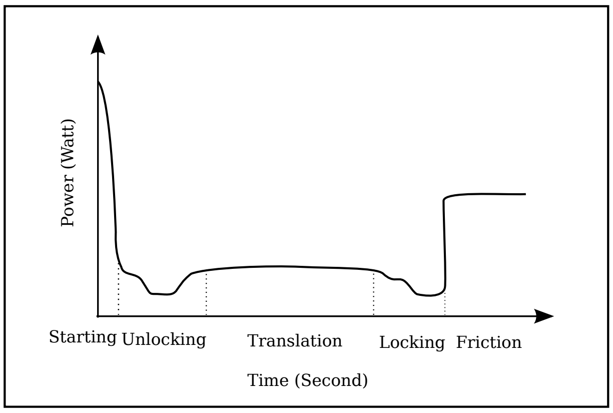


Figure 6.4: The curve of the electrical consumed power during a switch operation with the 5 electromechanical phases.

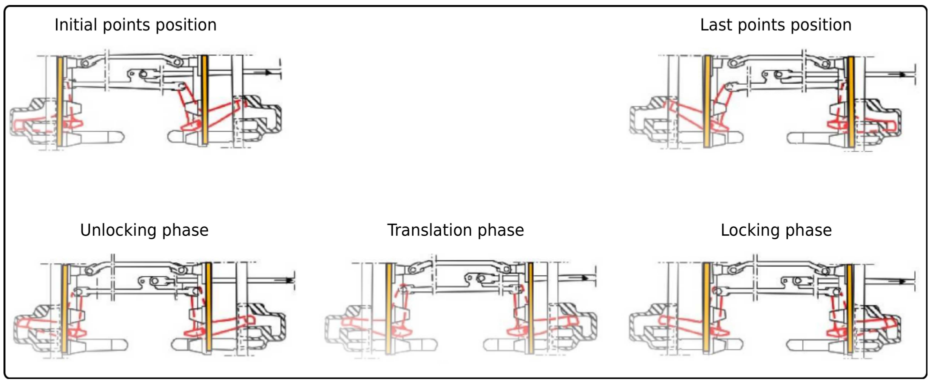


Figure 6.5: Physical motions involved in a switch operation.

6.3.3 The used database

A database of $n = 120$ real switch operation curves with three known states (classes) was used. Given by a railway expert, the considered classes of curves correspond to the different operating states of the switch mechanism:

- $g = 1$: no defect class;
- $g = 2$: minor defect class;
- $g = 3$: critical defect class.

The cardinal numbers of the three classes are $n_1 = 35$, $n_2 = 40$ and $n_3 = 45$ respectively. Figure 6.6 shows a curve example from each operating state.

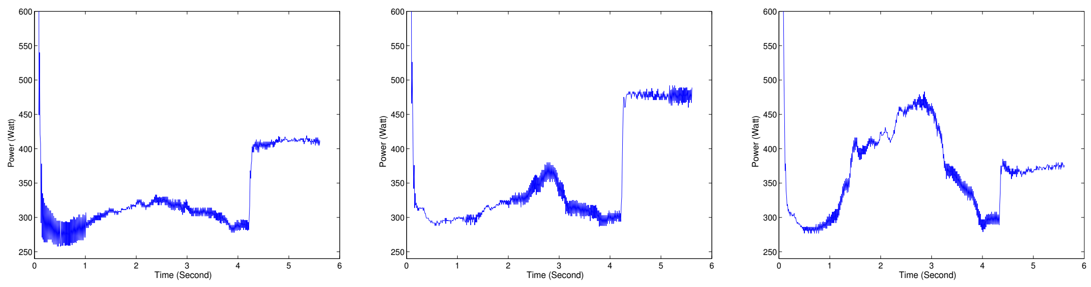


Figure 6.6: Example of curve corresponding to a normal switch operation without defect (a), a curve with a minor defect (b) and a curve with critical defect (c).

6.4 Adopted statistical pattern recognition methodology

In this section we briefly describe the global pattern recognition approach used in the diagnosis task, which results in a classification problem. It is composed of the following stages:

Data acquisition: This step is known as the sensors part in which the rough data (e.g, signals, images ...) are acquired (e.g., measured) through dedicated sensors.

Preprocessing: Preprocessings of the rough data are generally required: For example denoising, standardization, normalization and a feature extraction. Note that "feature extraction" is referred to the data representation task and not yet to "feature selection".

Feature selection: This step is concerned with the optimization of the representation space by applying for example, linear or nonlinear dimensionality reduction techniques. For the linear case, one can think about Principal Component Analysis (PCA) or Independent component analysis (ICA) for the unsupervised context. For the non-linear case, one can cite Kernel Principal Component Analysis (Kernel PCA), Multidimensional Scaling (MDS), Kohonen Maps (SOMs) or its generative version that is Generative Topographic Mapping (GTM), etc. These methods are applied on unlabeled data, so they are dedicated to unsupervised context. For supervised context we can cite linear methods like factor analysis (FA), Sequential Forward Selection (SFS), Partial Least Square Regression (PLS) or non linear methods as Curvilinear Component Analysis (CCA), relevance feature vector machine (RFVM), or potential support vector machine (P-SVM).

Classifier design: Given a training set of selected features (observations), the classification stage is concerned with the designing of a decision rule with respect to a chosen optimality criterion. The main questions concerning the classifier design stage are: the type of the approach (generative, discriminative,..), linear/non-linear separation, and the type of the criterion. In our study, we focus on generative classifiers in a maximum likelihood estimation framework.

Classifier evaluation: Finally, once the classifier is designed, the performance of the classifier has to be assessed, for example by computing the classification error rate on new data examples, in order to evaluate its generalization capabilities.

Figure 6.7 summarizes the various stages followed for the design of a pattern recognition system. Notice that these stages are not independent. On the contrary, they

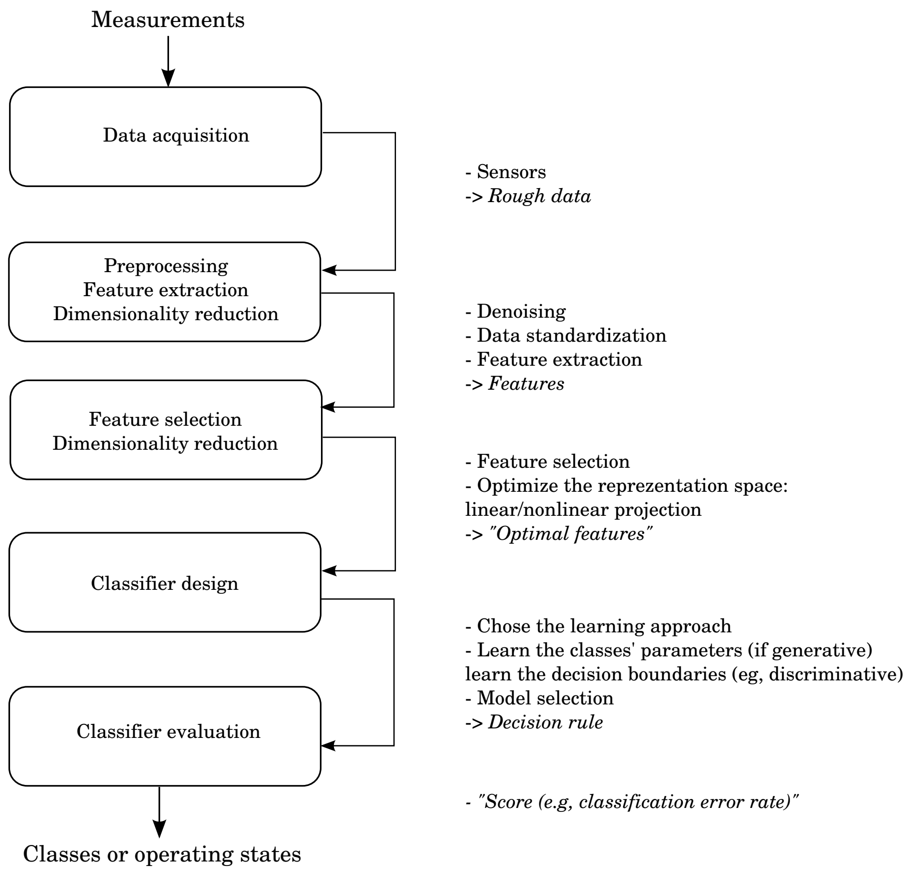
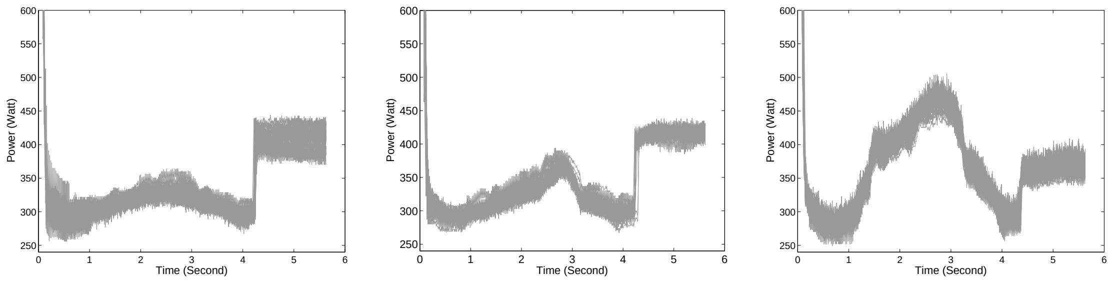


Figure 6.7: The basic stages involved in the design of a pattern recognition system.

are highly interrelated and, depending on the results, one may go back to redesign earlier stages in order to improve the overall performance. Furthermore, there are some methods that combine stages, for example, in our case, the approach seen in Chapter 3 involves a statistical learning at two stage. The first is concerned with learning the RHLP model for each curve, and the second of learning a mixture model (MDA) to perform curve classification. Some stages can also be removed, for example, as seen in Chapter 4, the classifier is directly built without feature extraction nor feature selection.

6.5 Classification of the switch operation curves

This section is dedicated to the classification of the curves issued from the switch operations. The three sets of curves corresponding to the three operating states of the switch are shown in Figure 6.8.



(a) (b) (c)

Figure 6.8: Sets of curves of electrical power consumed during various switch operations: 35 curves without defect (a), 40 curves with a minor defect (b) and 45 curves with critical defect (c).

The curves at this stage are assumed to be independent. We consider the RHLP model for a single curve (see Chapter 3) and for a set of curves (see Chapter 4).

6.5.1 Experimental setup

For the switch operation curves, the number of regression components is chosen in accordance with the number of electromechanical phases of a switch operation ($K = 5$) (see Figure 6.4 and Figure 6.5). The degree of the polynomial regression p was set to 3 which is appropriate for the different regime involved in the curves.

The two proposed approaches (RHLP-MDA and RHLP-MAP) are compared to the other alternative approaches.

Curve classification by RHLP-MDA

The first approach we use for curve classification is based of the pecific regression model with hidden logistic process (RHLP) presented in Chapter 3 (Chamroukhi et al., 2009c). Since a large amount of data composes each curve (each curve has 564 points), the dimension of the deduced representation space is too large for a classical pattern recognition task (eg, LDA, QDA, MDA, etc) in which the classification is performed in the feature space. Consequently, a feature extraction task is required to deliver a reduced input space that resumes information efficiently and allows for avoiding the curse of dimensionality problem (Bishop, 1995). This is achieved by applying the RHLP model to each switch operation curve. The model parameters for each curve are estimated by maximum likelihood via the EM algorithm implemented in algorithm 3.

The alternative methods considered for comparisons are the polynomial spline regression model (PSR) (Deboor, 1978), the HMM Regression model (HMMR) (Fridman, 1993) and the piecewise polynomial regression model (PWPR) (Bellman, 1961; Hébrail et al., 2010; McGee and Carleton, 1970; Stone, 1961). For each curve we apply the corresponding modeling approach, and the estimated model parameters are then used as the feature vector for that curve.

After the feature extraction step, the supervised learning of the different operating states (no defect, minor defect and critical defect) is then performed from a labeled collection of curves using Mixture Discriminant Analysis (MDA) (Hastie and Tibshirani, 1996). Based on the learned operating states' parameters, a new curve is then classified in the feature space by using the MAP rule. This two-fold classification approach was presented in section 3.4.1.

Curve classification by RHLP-MAP

In the second approach, we use the RHLP model for a set of curves to represent each class of curves of the switch mechanism through a single mean curve.

Each class of curves is summarized by finding "a mean" curve using the RHLP model for a set of curves presented in Chapter 4 (Chamroukhi et al., 2010). According to this approach, the parameters of each class of curves are directly learned in the space of curves by using algorithm 4. Based on the estimated class parameters, the curve classification is then directly performed in the space of curves by the MAP decision principle. Therefore, the feature extraction step is no longer required in this approach. The modeling and the classification tasks being directly performed in the space of curves.

The considered models for comparisons are the PWPR model (Bellman, 1961; Hébrail et al., 2010; McGee and Carleton, 1970; Stone, 1961) and the PSR model (Deboor, 1978; James and Hastie, 2001) for a set of curves. The HMMR model is not adapted for the case of a set of curves. Let us recall that under this Markov modeling, as seen in 2.6.6, the mean curve is computed from the observations through the posterior (smooth) probabilities. Therefore, an overall mean curve can not directly be produced in the case of a set of curves as the posterior probabilities differ from one curve to another. For all models, the last stage regarding the curve classification is performed by Functional Linear Discriminant Analysis (FLDA) approach as in (James and Hastie, 2001) (c.f., section 4.2.6).

6.5.2 Classification results

The classification results obtained with the two approaches are given in Table 6.1.

It can be seen that the performance of the PSR approach is significantly outperformed by the other approaches. This can be attributed to the fact that the curves shape, as they present various regime changes, are more complex to be modeled by splines. The classification results with the proposed RHLP approach followed by MDA, confirm that using the proposed RHLP approach for curve modeling, rather than spline

Modeling approach	Misclassification error rate (%)
PSR-MDA	13 ±(4.5)
PWPR-MDA	12 ±(1.7)
HMMR-MDA	9 ±(2.25)
RHLP-MDA	**4 ±(1.33)**
PSR-MAP	7.3 ± (4.36)
PWPR-MAP	1.82 ± (5.74)
RHLP-MAP	**1.67 ± (2.28)**

Table 6.1: Classification results for the switch operation curves.

regression, HMM regression or piecewise regression, is quite promising for the two-strategy curve classification. It can also be observed that, in this case, the HMMR approach is the more competitive with the RHLP approach. The results also indicate that the PWPR approach is significantly outperformed by the RHLP approach methods for the first two-stage strategy (MDA approaches) but is not significantly outperformed for the second approach (MAP approaches).

The conclusion that can be drawn from this experiment is that the RHLP approach is the most successful for the two strategies. Beyond the misclassification error, the modeling results presented below, clearly show the advantage of using the proposed RHLP model.

The modeling results obtained with the PWPR model for the curve examples presented in Figure 6.6 are shown on Figure 6.9. Figure 6.12 shows the modeling results obtained with the proposed RHLP model for the same curves presented in Figure 6.6. Figure 6.12 (top) shows the original curves (in gray) and the estimated polynomial regression components (in dashed and continuous lines). The polynomial components are represented in dashed and continuous lines. The part for which the kth polynomial model is active is represented in continuous line. Figure 6.12 (middle) shows the variation of the logistic probabilities $\pi_k(t)$ for the five polynomial components over time. Finally, the original curves and the estimated curves provided by the RHLP model are shown in Figure 6.12 (bottom).

It can be seen that the logistic process probabilities are closed to 1 when the kth regression model seems to be the best fit for the original curve. The different phases of the operation are identified through the logistic probabilities and are revealed to be accurate with respect to the actual phases of the switch operation. It can also be observed that the RHLP model provides an accurate model for each curve. Notice that the estimated curve is computed according to Equation (3.22). Compared to the PWPR approach, it can be observed that, when the transitions are very pronounced, as in the starting phase, the transition phases and the phases of the last switch operation (c.f., Figure 6.6(c)), the results are quasi-similar. However, for the other transitions, the RHLP model provides more accurate modeling results and ensure the continuity of the estimated curve.

For the second strategy, which is equivalent to use a model for a set of curves

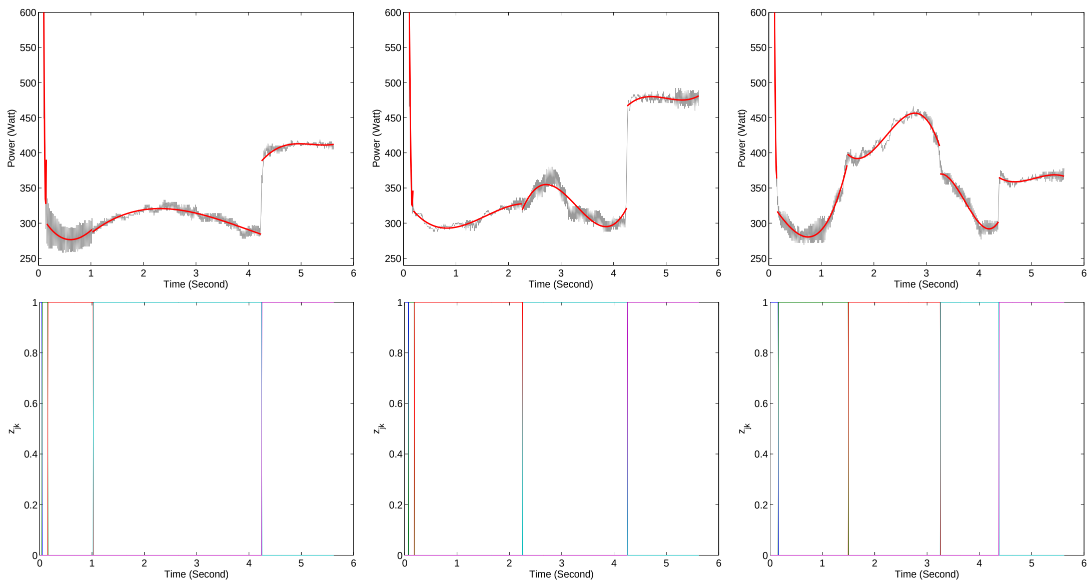


Figure 6.9: Results obtained with the piecewise regression model (PWPR) for a curve without defect (left), a curve with minor defect (middle) and a curve with critical defect (right) with the original curves and the estimated piecewise curves (top), and the corresponding segmentation (bottom).

followed by a MAP decision rule, Figure 6.11 shows the modeling results obtained with piecewise regression (PWPR) approach.

Figure 6.12 shows the switch operation curve, the estimated polynomial component of RHLP for each regime, the corresponding probabilities of the logistic process and the estimated mean curve for each class of curve. We see that the proposed method ensure, unlike the piecewise regression approach, the continuity of the estimated curves. In addition, it can be observed that, for the transitions corresponding to the first and the last phase, the changes are abrupt for the three classes. The estimated probabilities are therefore very close to 1, and the transitions are accurately estimated by the two approaches. On the other hand, when the regime are less pronounced, in particular for the middle phases of the first class, the flexibility of the logistic process allows for providing an adapted model estimation through smooth logistic probabilities. The estimated phases have also been found realistic regarding the true phases involved in the real switch operations.

Until now we were interested in the static case in which the used models are learned from independent curves. For the dynamical case, we will be interested in modeling the dynamical aspect of the switch mechanism degradation from a curve sequence by using the switching AR models presented in Chapter 5.

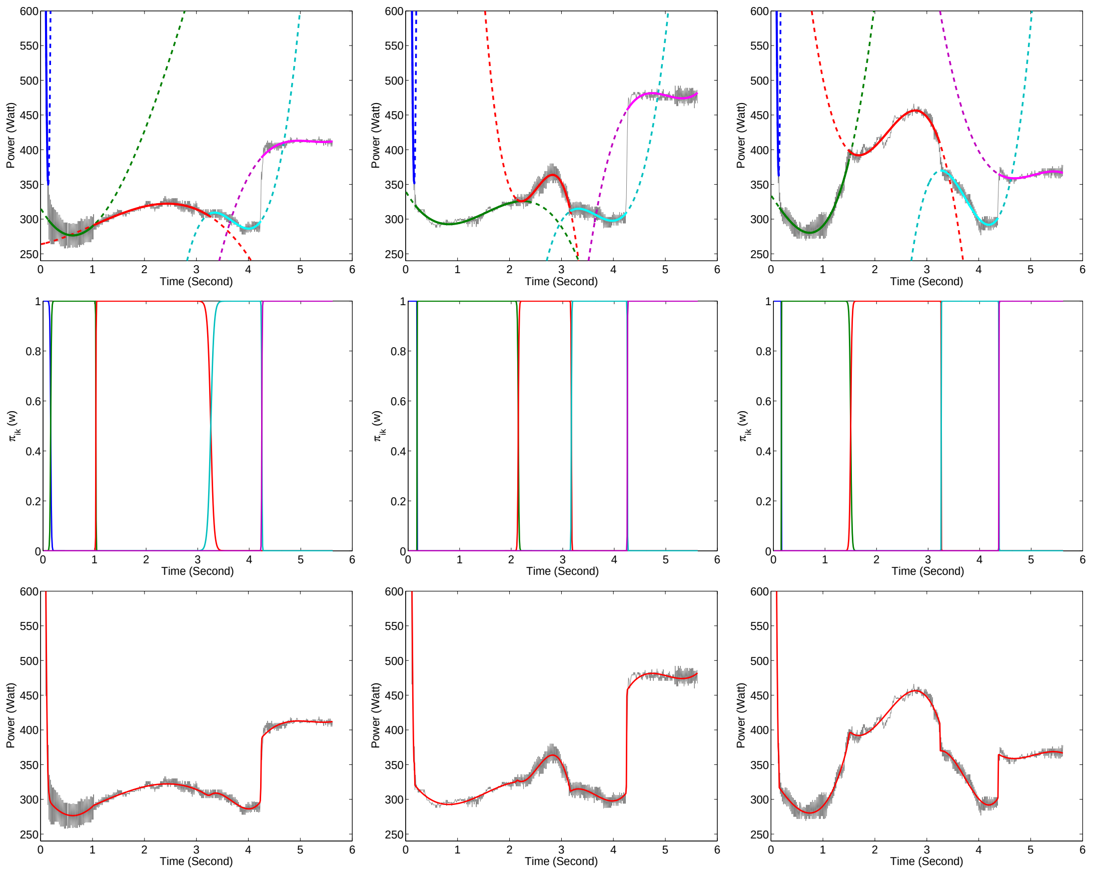


Figure 6.10: Results obtained with the proposed approach (RHLP) for a curve without defect (left), a curve with minor defect (middle) and a curve with critical defect (right) with the original curves and the estimated polynomial regression components of RHLP (top), the corresponding probabilities of the logistic process over time $\pi_k(t)$ $(k = 1, \ldots, 5)$ (middle) and the original curves and the estimated curves (bottom).

6.6 Dynamical monitoring of the switch mechanism

6.6.1 Experimental setup

The dynamical modeling approach is performed by means of the two-step strategy detailed in section 5.2. Let us recall that this approach consists of a feature extraction from the sequence of switch operation curves, followed by a dynamical modeling of the sequence of extracted features by using switching autoregressive models. The feature extraction from each switch operation curve is performed by fitting the RHLP model to each curve. Based on the extracted features, the evolution of the switch operating state over time is modeled using the proposed multivariate specific autoregressive models. We have tested the multivariate autoregressive model governed by a hidden logistic process (ARHLP) and the multivariate autoregressive non-homogeneous Hidden Markov Model

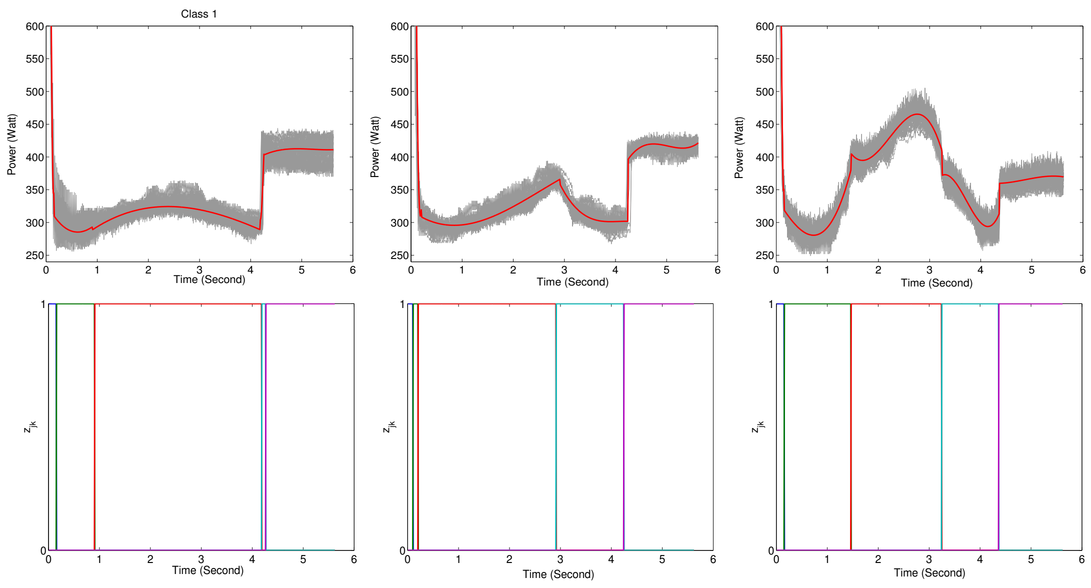


Figure 6.11: The three classes of switch operation curves and the corresponding estimated curve (top) and the segmentation (bottom) obtained with the piecewise polynomial regression model.

(AR-NHMM). For each model, the hidden process automatically controls the switching from one autoregressive model to another (among K models) over time. We assume that each autoregressive model is associated with an operating state of the system. The true sequence of the operating states is shown in Figure 6.13.

In the following section we give the dynamical modeling results obtained with the ARHLP and the AR-NHMM models.

6.6.2 Obtained results

In this section we report the modeling results obtained by the batch version of the ARHLP and the AR-NHMM approaches for the dynamical estimation of the switch operating states.

Figure 6.14 shows the logistic probabilities (prediction probabilities) obtained with the ARHLP model, and Figure 6.15 shows the estimated MAP sequence.

The estimated sequence is obtained by maximizing, at each time step, the posterior class probability. Let us recall that the prediction probabilities and the posterior probabilities are computed according to Equation (5.41) and Equation (5.39), respectively. The class prediction error rate in this case is equal to 10.92% and the error rate between the true sequence and the estimated MAP sequence is equal to 9.24 %.

Figure 6.16 shows the prediction probabilities obtained with the Autoregressive non-homogeneous HMM and the estimated state sequence is shown in Figure 6.17. In this case of the AR-NH-HMM model, the prediction error rate is equal to 9.24 % and

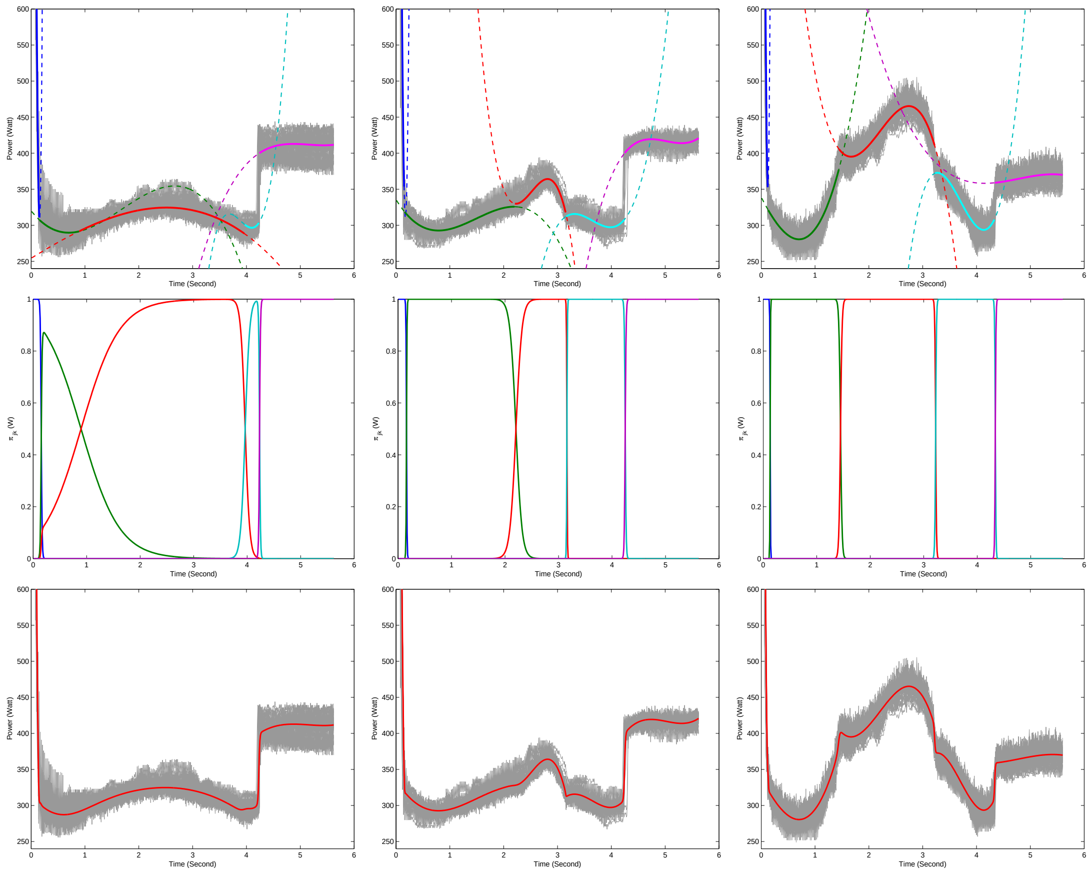


Figure 6.12: Results obtained with the proposed RHLP model for the class of curves without defect (left), with minor defect (middle) and with critical defect (right), with the original curves and the estimated polynomial regression components (top), the corresponding probabilities of the logistic process over time $\pi_k(t)$ ($k = 1, \ldots, 5$) (middle) and the original curves and the estimated curves (bottom).

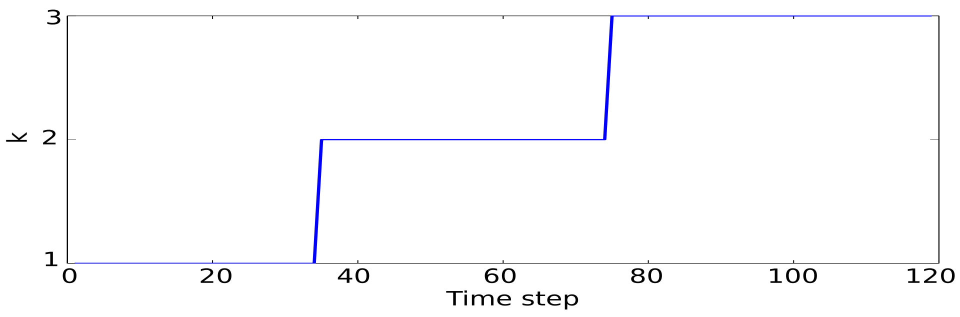


Figure 6.13: True state sequence of switch operation curves (120 curves).

the error rate between the true sequence and the estimated sequence is equal is equal to 8.40 %.

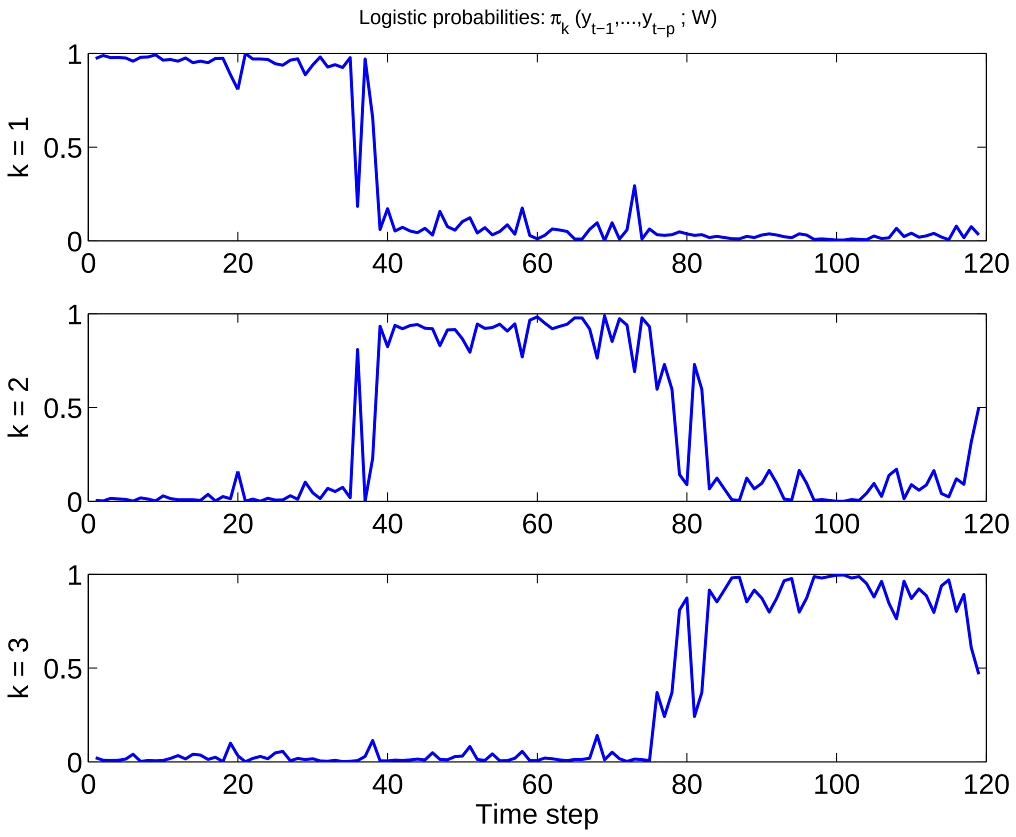


Figure 6.14: Prediction probabilities over time for the switch operating states obtained with the ARHLP model ($p = 1$) from the curve sequence.

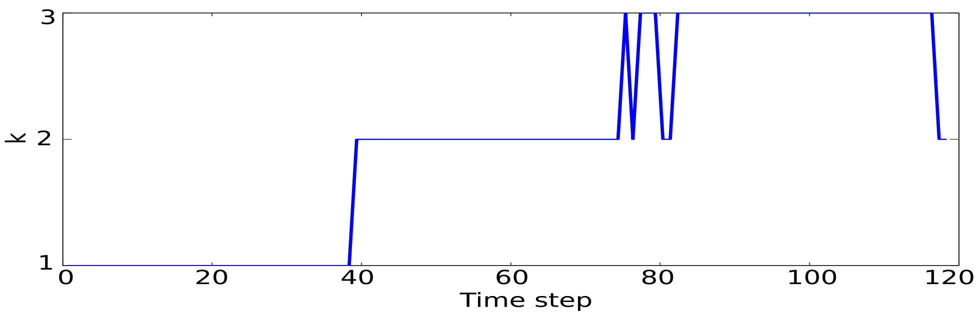


Figure 6.15: State sequence obtained with the ARHLP model from the switch operation curves.

Figure 6.18 illustrates the fact that the transition matrix $\mathbf{A} = (\mathbf{A}_{ij})$ is a function of time. In this matrix plot, the subplot (i, j) represnts the temporal evolution of $\mathbf{A}_{ij}$.

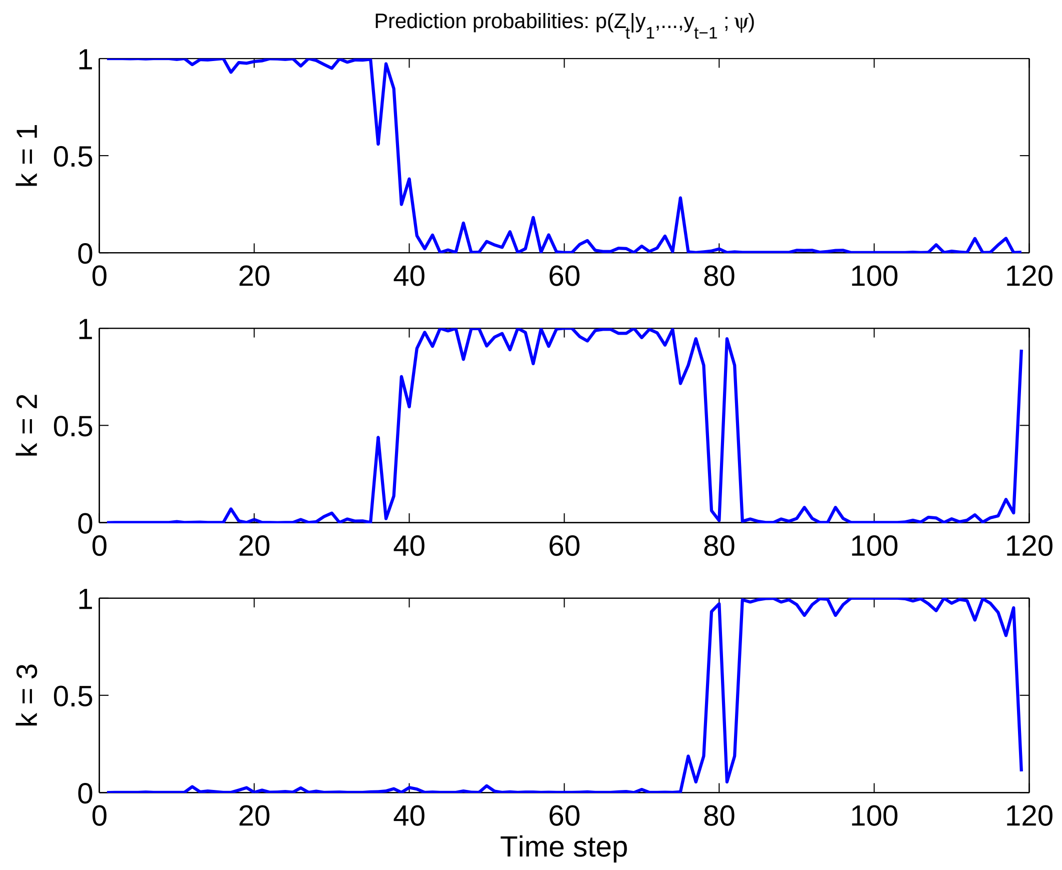


Figure 6.16: Prediction probabilities over time for the switch operating states obtained with the AR-NH-HMM model from the curve sequence.

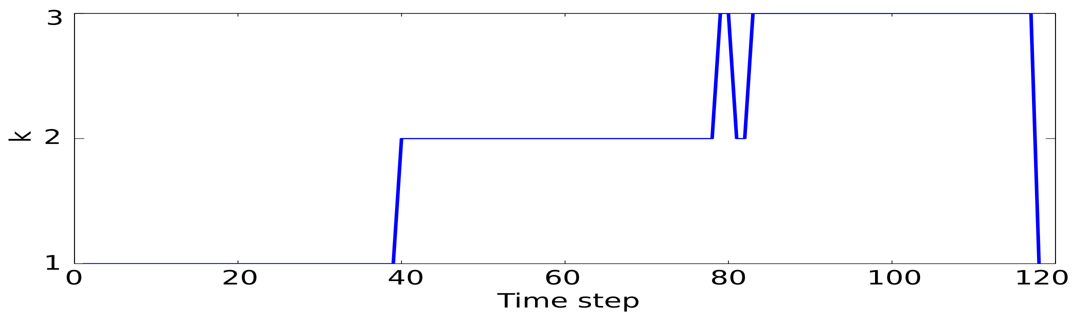


Figure 6.17: State sequence obtained with the AR-NH-HMM model from the switch operation curves.

6.7 Summary

This chapter focused on modeling and classifying switch operation curves in order to diagnose real cases of railway switches. We applied the methodologies developed in

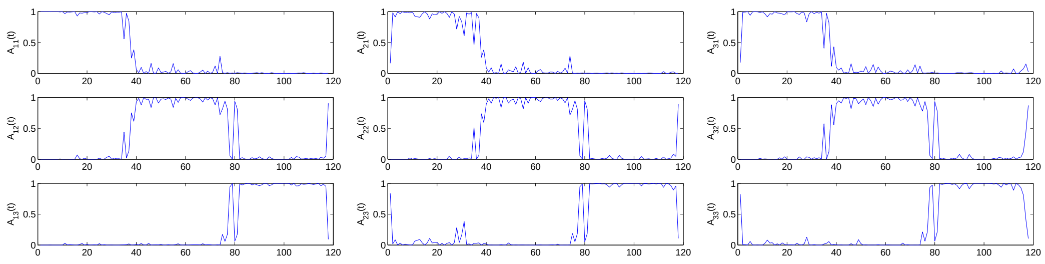


Figure 6.18: Time-varying transition probabilities for the AR-NH-HMM model estimated from the switch operation curve sequence.

previous chapters to implement the diagnosis task.

First, we were interested in the static (or stationary) case, where the aim is to make decisions about the operating state independently of the evolution of the system over time. The two-step RHLP-MDA approach (see Chapter 3) and the direct RHLP-MAP classification approach (see Chapter 4) show very good performance for identifying the actual operating states based on a labeled training set of switch operation curves.

Second, we focused on the non-stationary case by modeling the dynamical degradation of the system from a curve sequence. The approaches consist of specific autoregressive (AR) models governed by hidden processes. The proposed approaches (see Chapter 5) show that they can capture dynamical behavior as well as make predictions and decisions over time. An experimental study was performed to assess the different approaches.

Chapter 7

Conclusions and future directions

Conclusions

In this thesis, we introduced novel methodologies for curve modeling, classification, clustering and tracking. The developed approaches are particularly suited to curves presenting smooth and/or abrupt regime changes.

In Chapter 2, we first presented a new probabilistic approach for curve modeling that is based on a dynamical regression model by incorporating a hidden logistic process (RHLP). The proposed model provides an accurate representation of the curve with respect to the involved dynamical regimes and is very useful from a curve dimensionality reduction perspective. We then presented a two-fold approach for curve clustering and classification using the RHLP model. It relies on external models: in this case, mixture model-based clustering in the unsupervised context and MDA in the supervised context. In this approach, classification is performed in the space of curve descriptors. Experiments on simulated data showed the effectiveness of the proposed approach in terms of curve modeling and classification as compared with other alternative approaches, including the piecewise regression and the HMM regression.

In Chapter 4, we extended this probabilistic RHLP modeling approach to model a set of homogeneous curves. We then proposed a mixture approach that integrates the RHLP model to represent a set of heterogeneous curves, that is, the MixRHLP model, which is naturally tailored to address the problem of complex shaped classes of curves. The classification using the RHLP and the MixRHLP models is directly performed in the space of curves and can be associated with the Functional Data Analysis framework. Experiments conducted on simulated sets of curves demonstrated the relevance of the proposed approaches in terms of curve modeling and classification as compared to other alternatives, including spline regression and piecewise regression. In addition, the experiments with simulated data showed that the direct classification approach seems superior to the two-fold approach.

Beyond curve modeling and classification problems, the developed RHLP approach automatically identifies the underlying regimes, and so it should be generally useful for the problem of (online) change-point detection as well as for signal denoising and segmentation.

In Chapter 5, we further presented dynamical modeling approaches for learning from a curve sequence using a two-step strategy. The first stage consists in feature extraction from the curves and the second one involves specific probabilistic models to model the formed sequence of multidimensional data. The formulation of the proposed switching autoregressive model is, on the one hand, appropriate for modeling a time-varying state; on the other hand, it is suitable for addressing the problem of missing information. We also presented corresponding learning techniques in a batch mode and in an online mode. The models were tested on multidimensional data, which are assumed to be extracted from curves. A second experimental study performed on simulated curve sequence showed the performances of the proposed approaches.

Finally, the real-world application that deals with the problem of railway switch diagnosis in both static and dynamic frameworks demonstrates the practical use of the ideas introduced in this thesis.

Future directions

A current work evaluates the online learning approaches for modeling curve sequences.

Future works may develop an extension of the RHLP model in order to derive a direct approach that enables dynamical modeling from curve sequences.

Although inspired by the diagnosis task for which the whole curve is available, the RHLP model for curves may be applied to the particular problem of online change-point detection. This problem could be solved straightforwardly with an online EM by relying on the EM formulations presented in Chapter 5. Additionally, and perhaps more interesting, RHLP models in their current formulations should be very promising for the segmentation and classification of the array Comparative Genomic Hybridization (CGH) profiles (see for example (Bleakley and Vert, 2009)).

In addition, a semi-supervised learning context for the RHLP model for curve modeling/segmentation is also straightforward.

Inspired by the diagnosis application, the second step in the two-fold strategy presented in Chapter 5, focused on the discrete state assumption. It can however be extended to a continuous state model (e.g., a Kalman Filtering approach). A problem to be addressed in this case is the optimization of the state space dimension.

A Bayesian framework could also be adopted through assuming a prior distribution on the RHLP model parameters. However, for the application, I think that a MAP estimation framework providing a distribution for the “mean curve” rather than a fixed curve, may not be of interest for railway managers team, in the sense that they are more familiar with a “deterministic” representation the proper functioning state, through a mean curve model.

While my interests in this research lied in developing machine learning approaches inspired by the diagnosis application domain, I look forward to work in other areas of statistical learning and computer science where I would be able to contribute, especially the areas of artificial intelligence (robotics) and bioinformatics.

Appendix A

Appendix A

A.1 EM algorithm: updating the proportions of a mixture model using Lagrange multipliers

Consider the problem of finding the maximum of the function

$$Q_\pi(\pi_1,\ldots,\pi_K,\mathbf{\Psi}^{(q)}) = \sum_{i=1}^{n}\sum_{k=1}^{K}\tau_{ik}^{(q)}\log\pi_k$$

given by Equation (2.12), with respect to the mixing proportions $(\pi_1,\ldots,\pi_K)$ subject to the constraint $\sum_{k=1}^{K}\pi_k = 1$. To perform this constrained maximization, we introduce the Lagrange multiplier λ such that the resulting Lagrangian function is given by:

$$L(\pi_1,\ldots,\pi_K) = \sum_{i=1}^{n}\sum_{k=1}^{K}\tau_{ik}^{(q)}\log\pi_k + \lambda(1-\sum_{k=1}^{K}\pi_k). \tag{A.1}$$

Taking the derivatives of the Lagrangian with respect to π_k for $k = 1,\ldots,K$ we obtain:

$$\frac{\partial L(\pi_1,\ldots,\pi_K)}{\partial\pi_k} = \frac{\sum_{i=1}^{n}\tau_{ik}^{(q)}}{\pi_k} - \lambda, \quad \forall k \in \{1,\ldots,K\}. \tag{A.2}$$

Then, setting these derivatives to zero yields:

$$\frac{\sum_{i=1}^{n}\tau_{ik}^{(q)}}{\pi_k} = \lambda, \quad \forall k \in \{1,\ldots,K\}. \tag{A.3}$$

By multiplying each hand side of (A.3) by π_k $(k = 1,\ldots,K)$ and summing over k we get

$$\sum_{k=1}^{K}\frac{\pi_k \times \sum_{i=1}^{n}\tau_{ik}^{(q)}}{\pi_k} = \sum_{k=1}^{K}\lambda\times\pi_k \tag{A.4}$$

which implies that $\lambda = n$. Finally, from (A.3) we get the updating formula for the mixing proportions π_k's, that is

$$\pi_k^{(q+1)} = \frac{\sum_{i=1}^{n} \tau_{ik}^{(q)}}{\lambda} = \frac{\sum_{i=1}^{n} \tau_{ik}^{(q)}}{n}, \quad \forall k \in \{1, \ldots, K\}. \tag{A.5}$$

A.2 Calculation of the derivative of the multi-class logistic function

The partial derivative of the logistic function

$$\pi_k(\mathbf{x}_i; \mathbf{w}) = \frac{\exp(\boldsymbol{w}_k^T \mathbf{x}_i)}{\sum_{h=1}^{K} \exp(\boldsymbol{w}_h^T \mathbf{x}_i)} \quad ; \quad (k = 1, \ldots, K) \tag{A.6}$$

with respect to $\boldsymbol{w}_\ell$ $(\ell = 1, \ldots, K)$ is given by

$$\begin{aligned}
\frac{\partial \pi_k(\mathbf{x}_i; \boldsymbol{w})}{\partial \boldsymbol{w}_\ell} &= \frac{\partial \frac{\exp(\boldsymbol{w}_k^T \mathbf{x}_i)}{\sum_{h=1}^{K} \exp(\boldsymbol{w}_h^T \mathbf{x}_i)}}{\partial \boldsymbol{w}_\ell} \\
&= \frac{\frac{\partial \exp(\boldsymbol{w}_k^T \mathbf{x}_i)}{\partial \boldsymbol{w}_\ell} \sum_{h=1}^{K} \exp(\boldsymbol{w}_h^T \mathbf{x}_i) - \exp(\boldsymbol{w}_k^T \mathbf{x}_i) \frac{\partial \sum_{h=1}^{K} \exp(\boldsymbol{w}_h^T \mathbf{x}_i)}{\partial \boldsymbol{w}_\ell}}{(\sum_{h=1}^{K} \exp(\boldsymbol{w}_h^T \mathbf{x}_i))^2} \\
&= \frac{\delta_{k\ell} \mathbf{x}_i \exp(\boldsymbol{w}_k^T \mathbf{x}_i) \sum_{h=1}^{K} \exp(\boldsymbol{w}_h^T \mathbf{x}_i) - \exp(\boldsymbol{w}_k^T \mathbf{x}_i) \mathbf{x}_i \exp(\boldsymbol{w}_\ell^T \mathbf{x}_i)}{(\sum_{h=1}^{K} \exp(\boldsymbol{w}_h^T \mathbf{x}_i))^2} \\
&= \delta_{k\ell} \frac{\exp(\boldsymbol{w}_k^T \mathbf{x}_i)}{\sum_{h=1}^{K} \exp(\boldsymbol{w}_h^T \mathbf{x}_i)} \mathbf{x}_i - \frac{\exp(\boldsymbol{w}_k^T \mathbf{x}_i) \exp(\boldsymbol{w}_\ell^T \mathbf{x}_i)}{(\sum_{h=1}^{K} \exp(\boldsymbol{w}_h^T \mathbf{x}_i))^2} \mathbf{x}_i \\
&= \delta_{k\ell} \pi_k(\mathbf{x}_i; \boldsymbol{w}) \mathbf{x}_i - \pi_k(\mathbf{x}_i; \boldsymbol{w}) \pi_\ell(\mathbf{x}_i; \boldsymbol{w}) \mathbf{x}_i \\
&= \pi_k(\mathbf{x}_i; \boldsymbol{w}) \left(\delta_{k\ell} - \pi_\ell(\mathbf{x}_i; \boldsymbol{w})\right) \mathbf{x}_i
\end{aligned} \tag{A.7}$$

where $\delta_{k\ell}$ is the Kronecker delta such that $\delta_{k\ell} = 1$ if $k = \ell$ et $\delta_{k\ell} = 0$ otherwise, and in the third step we used the fact that $\frac{\partial (\boldsymbol{w}_k^T \mathbf{x}_i)}{\partial \boldsymbol{w}_k} = \mathbf{x}_i$. Finally we have

$$\frac{\partial \pi_k(\mathbf{x}_i; \boldsymbol{w})}{\partial \boldsymbol{w}_\ell} = \pi_k(\mathbf{x}_i; \boldsymbol{w}) \left(\delta_{k\ell} - \pi_\ell(\mathbf{x}_i; \boldsymbol{w})\right) \mathbf{x}_i \tag{A.8}$$

and similarly we get

$$\frac{\partial \pi_k(\mathbf{x}_i; \boldsymbol{w})}{\partial \boldsymbol{w}_\ell^T} = \pi_k(\mathbf{x}_i; \boldsymbol{w}) \left(\delta_{k\ell} - \pi_\ell(\mathbf{x}_i; \boldsymbol{w})\right) \mathbf{x}_i^T. \tag{A.9}$$

A.3 Gradient and Hessian for the multi-class logistic regression model

The function optimized for the multi-class logistic model model is given by Equation (2.53). To calculate the gradient of this function, since from (A.2) we have

$$\frac{\partial \pi_g(\mathbf{x}_i;\mathbf{w})}{\partial \boldsymbol{w}_h} = \pi_g(\mathbf{x}_i;\mathbf{w})\left(\delta_{gh} - \pi_h(\mathbf{x}_i;\mathbf{w})\right)\mathbf{x}_i. \tag{A.10}$$

for $g, h = 1, \ldots, G-1$ where δ_{gh} is the Kronecker delta such that $\delta_{gh} = 1$ if $g = h$ and $\delta_{gh} = 0$ otherwise, the component gradient vector $\frac{\partial \mathcal{L}(\mathbf{w})}{\partial \boldsymbol{w}_h}$ $(h = 1, \ldots, G-1)$ is therefore given by:

$$\begin{aligned}
\frac{\partial \mathcal{L}(\mathbf{w})}{\partial \boldsymbol{w}_h} &= \sum_{i=1}^{n}\sum_{g=1}^{G} y_{ig}\frac{\partial \log \pi_g(\mathbf{x}_i;\mathbf{w})}{\partial \boldsymbol{w}_h} \\
&= \sum_{i=1}^{n}\sum_{g=1}^{G} y_{ig}\frac{1}{\pi_g(\mathbf{x}_i;\mathbf{w})}\frac{\partial \pi_g(\mathbf{x}_i;\mathbf{w})}{\partial \boldsymbol{w}_h} \\
&= \sum_{i=1}^{n}\sum_{g=1}^{G} y_{ig}\frac{1}{\pi_g(\mathbf{x}_i;\mathbf{w})}\pi_g(\mathbf{x}_i;\mathbf{w})\left(\delta_{gh} - \pi_h(\mathbf{x}_i;\mathbf{w})\right)\mathbf{x}_i \\
&= \sum_{i=1}^{n}\sum_{g=1}^{G} y_{ig}\left(\delta_{gh} - \pi_h(\mathbf{x}_i;\mathbf{w})\right)\mathbf{x}_i \\
&= \sum_{i=1}^{n}\sum_{g=1}^{G} \left(y_{ig}\delta_{gh} - y_{ig}\pi_h(\mathbf{x}_i;\mathbf{w})\right)\mathbf{x}_i \\
&= \sum_{i=1}^{n}\left(\sum_{g=1}^{G} y_{ig}\delta_{gh} - \pi_h(\mathbf{x}_i;\mathbf{w})\sum_{g=1}^{G} y_{ig}\right)\mathbf{x}_i \\
&= \sum_{i=1}^{n}\big(y_{ih} - \pi_h(\mathbf{x}_i;\mathbf{w})\big)\mathbf{x}_i
\end{aligned} \tag{A.11}$$

where before the last step we have used the fact that $\sum_{g=1}^{G} y_{ig} = 1$.

The Hessian matrix is composed of $(G-1)\times(G-1)$ block matrices where each block matrix is of dimension $(d+1) \times (d+1)$ and is given by:

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}(\mathbf{w})}{\partial \boldsymbol{w}_h \partial \boldsymbol{w}_k^T} &= \frac{\partial \frac{\partial \mathcal{L}(\mathbf{w})}{\partial \boldsymbol{w}_h}}{\partial \boldsymbol{w}_k^T} = \frac{\partial \sum_{i=1}^{n} \left(y_{ih} - \pi_h(\mathbf{x}_i; \mathbf{w})\right) \mathbf{x}_i}{\partial \boldsymbol{w}_k^T} \\
&= -\sum_{i=1}^{n} \mathbf{x}_i \frac{\partial \pi_h(\mathbf{x}_i; \mathbf{w})}{\partial \boldsymbol{w}_k^T} \\
&= -\sum_{i=1}^{n} \mathbf{x}_i \pi_h(\mathbf{x}_i; \mathbf{w}) \left(\delta_{hk} - \pi_k(\mathbf{x}_i; \mathbf{w})\right) \mathbf{x}_i^T \\
&= -\sum_{i=1}^{n} \pi_h(\mathbf{x}_i; \mathbf{w}) \left(\delta_{hk} - \pi_k(\mathbf{x}_i; \mathbf{w})\right) \mathbf{x}_i \mathbf{x}_i^T \qquad \text{(A.12)}
\end{aligned}$$

A.4 Forward-Backward algorithm for an HMM

Given a training set of sequential data $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_n)$ The Forward-Backward algorithm (Baum et al., 1970; Rabiner, 1989) is the recursive procedure used to compute the expected conditional distribution for an HMM parametrized by $\mathbf{\Psi} = (\pi, A, \boldsymbol{\theta})$ at the E-step of the EM (Baum-Welch) algorithm. Here we consider an HMM with continuous emission probabilities, and more particularly with Gaussian emission probabilities parametrized by $\boldsymbol{\theta}$. The Forward-Backward probabilities which can be denoted by α_{tk} and β_{tk} respectively, are defined as follows:

$$\alpha_{tk} = p(\mathbf{y}_1, \ldots, \mathbf{y}_t, z_t = k; \mathbf{\Psi}), \qquad \text{(A.13)}$$

which represents the probability of observing the partial sequence $(\mathbf{y}_1, \ldots, \mathbf{y}_t)$ and ending with the state k at the time step t, and

$$\beta_{tk} = p(\mathbf{y}_{t+1}, \ldots, \mathbf{y}_1 | z_t = k; \mathbf{\Psi}) \qquad \text{(A.14)}$$

is the probability of observing the rest of the sequence $(\mathbf{y}_{t+1}, \ldots, \mathbf{y}_1)$ knowing that the system starts from the stat k at the time step t. Note that the log-likelihood of the model parameters can be computed from the forward probabilities by

$$\log p(\mathbf{Y}; \mathbf{\Psi}) = \log \sum_{k=1}^{K} \alpha_{nk}.$$

The recursive calculation of these probabilities is performed as follows (Baum et al., 1970; Rabiner, 1989). We have

$$\begin{aligned}
\alpha_{tk} &= p(\mathbf{y}_1,\ldots,\mathbf{y}_t, z_t = k;\mathbf{\Psi})\\
&= p(\mathbf{y}_1,\ldots,\mathbf{y}_t|z_t = k;\mathbf{\Psi})p(z_t = k;\mathbf{\Psi})\\
&= p(\mathbf{y}_1,\ldots,\mathbf{y}_{t-1}|z_t = k;\mathbf{\Psi})p(\mathbf{y}_t|z_t = k;\mathbf{\Psi})p(z_t = k;\mathbf{\Psi})\\
&= p(\mathbf{y}_1,\ldots,\mathbf{y}_{t-1}, z_t = k;\mathbf{\Psi})p(\mathbf{y}_t|z_t = k;\mathbf{\Psi})\\
&= \sum_{\ell=1}^{K} p(\mathbf{y}_1,\ldots,\mathbf{y}_{t-1}, z_{t-1} = \ell, z_t = k;\mathbf{\Psi})p(\mathbf{y}_t|z_t = k;\mathbf{\Psi})\\
&= \sum_{\ell=1}^{K} p(\mathbf{y}_1,\ldots,\mathbf{y}_{t-1}|z_{t-1} = \ell, z_t = k;\mathbf{\Psi})p(z_t = k, z_{t-1} = \ell;\mathbf{\Psi})p(\mathbf{y}_t|z_t = k;\mathbf{\Psi})\\
&= \sum_{\ell=1}^{K} p(\mathbf{y}_1,\ldots,\mathbf{y}_{t-1}|z_{t-1} = \ell;\mathbf{\Psi})p(z_t = k|z_{t-1} = \ell;\mathbf{\Psi})p(z_{t-1} = \ell;\mathbf{\Psi})\\
&\qquad \times p(\mathbf{y}_t|z_t = k;\mathbf{\Psi})\\
&= \sum_{\ell=1}^{K} p(\mathbf{y}_1,\ldots,\mathbf{y}_{t-1}, z_{t-1} = \ell;\mathbf{\Psi})p(z_t = k|z_{t-1} = \ell;\mathbf{\Psi})p(\mathbf{y}_t|z_t = k;\mathbf{\Psi})\\
&= \Big[\sum_{\ell=1}^{K} \alpha_{(t-1)\ell} A_{\ell k}\Big] p(\mathbf{y}_t|z_t = k;\mathbf{\Psi}) \qquad \text{(A.15)}
\end{aligned}$$

and

$$\begin{aligned}
\beta_{t\ell} &= p(\mathbf{y}_{t+1},\ldots,\mathbf{y}_n|z_t = \ell;\mathbf{\Psi})\\
&= \sum_{k=1}^{K} p(\mathbf{y}_{t+1},\ldots,\mathbf{y}_n, z_{t+1} = k|z_t = \ell;\mathbf{\Psi})\\
&= \sum_{k=1}^{K} p(\mathbf{y}_{t+1},\ldots,\mathbf{y}_n|z_{t+1} = k, z_t = \ell;\mathbf{\Psi})p(z_{t+1} = k|z_t = \ell;\mathbf{\Psi})\\
&= \sum_{k=1}^{K} p(\mathbf{y}_{t+2},\ldots,\mathbf{y}_n|z_{t+1} = k, z_t = \ell;\mathbf{\Psi})p(z_{t+1} = k|z_t = \ell;\mathbf{\Psi})p(\mathbf{y}_{t+1}|z_{t+1} = k;\mathbf{\Psi})\\
&= \sum_{k=1}^{K} p(\mathbf{y}_{t+2},\ldots,\mathbf{y}_n|z_{t+1} = k;\mathbf{\Psi})p(z_{t+1} = k|z_t = \ell;\mathbf{\Psi})p(\mathbf{y}_{t+1}|z_{t+1} = k;\mathbf{\Psi})\\
&= \sum_{k=1}^{K} \beta_{(t+1)k} A_{\ell k} p(\mathbf{y}_{t+1}|z_{t+1} = k;\mathbf{\Psi}). \qquad \text{(A.16)}
\end{aligned}$$

Therefore, the computation of these quantities is performed by the Forward Backward procedure. For all $\ell, k = 1,\ldots,K$:

Forward procedure

- $\alpha_{1k} = p(\mathbf{y}_1, z_1 = 1;\mathbf{\Psi}) = p(z_1 = 1)p(\mathbf{y}_1|z_1 = 1;\boldsymbol{\theta}) = \pi_k p(\mathbf{y}_1|z_1 = k;\boldsymbol{\theta})$ for $t = 1$,

- $\alpha_{tk} = [\sum_{\ell=1}^{K} \alpha_{(t-1)\ell} A_{\ell k}] p(\mathbf{y}_t | z_t = k; \boldsymbol{\Psi}) \quad \forall\ t = 2, \ldots, n.$

Backward procedure

- $\beta_{nk} = 1$ for $t = n$,
- $\beta_{t\ell} = \sum_{k=1}^{K} \beta_{(t+1)k} A_{\ell k} p(\mathbf{y}_{t+1} | z_{t+1} = k; \boldsymbol{\Psi}) \quad \forall\ t = n-1, \ldots, 1.$

A.5 Posterior probabilities for an HMM

The posterior probability of the state k at time t given the whole sequence of observations $\mathbf{Y}$ and a model parameters $\boldsymbol{\Psi}$ is computed from the Forward-Backward and is given by

$$\begin{aligned}
\tau_{tk} &= p(z_t = k | \mathbf{Y}; \boldsymbol{\Psi}) \\
&= \frac{p(\mathbf{Y}, z_t = k; \boldsymbol{\Psi})}{p(\mathbf{Y}; \boldsymbol{\Psi})} \\
&= \frac{p(\mathbf{Y} | z_t = k; \boldsymbol{\Psi}) p(z_t = k; \boldsymbol{\Psi})}{\sum_{l=1}^{K} p(\mathbf{Y} | z_t = l; \boldsymbol{\Psi}) p(z_t = l; \boldsymbol{\Psi})} \\
&= \frac{p(\mathbf{y}_1, \ldots, \mathbf{y}_t | z_t = k; \boldsymbol{\Psi}) p(\mathbf{y}_{t+1}, \ldots, \mathbf{y}_n | z_t = k; \boldsymbol{\Psi}) p(z_t = k; \boldsymbol{\Psi})}{\sum_{l=1}^{K} p(\mathbf{y}_1, \ldots, \mathbf{y}_t | z_t = l; \boldsymbol{\Psi}) p(\mathbf{y}_{t+1}, \ldots, \mathbf{y}_n | z_t = l; \boldsymbol{\Psi}) p(z_t = l; \boldsymbol{\Psi})} \\
&= \frac{p(\mathbf{y}_1, \ldots, \mathbf{y}_t, z_t = k; \boldsymbol{\Psi}) p(\mathbf{y}_{t+1}, \ldots, \mathbf{y}_n | z_t = k; \boldsymbol{\Psi})}{\sum_{l=1}^{K} p(\mathbf{y}_1, \ldots, \mathbf{y}_t, z_t = l; \boldsymbol{\Psi}) p(\mathbf{y}_{t+1}, \ldots, \mathbf{y}_n | z_t = l; \boldsymbol{\Psi})} \\
&= \frac{\alpha_{tk} \beta_{tk}}{\sum_{l=1}^{K} \alpha_{tl} \beta_{tl}}.
\end{aligned} \tag{A.17}$$

The joint posterior probabilities of the state k at time t and the state ℓ at time $t-1$ given the whole sequence of observations are given by

$$\begin{aligned}
\xi_{t\ell k} &= p(z_t = k, z_{t-1} = \ell | \mathbf{Y}; \boldsymbol{\Psi}) \\
&= \frac{p(z_t = k, z_{t-1} = \ell, \mathbf{Y}; \boldsymbol{\Psi})}{p(\mathbf{Y}; \boldsymbol{\Psi})} \\
&= \frac{p(z_t = k, z_{t-1} = \ell, \mathbf{Y}; \boldsymbol{\Psi})}{\sum_{\ell=1}^{K} \sum_{k=1}^{K} p(z_t = k, z_{t-1} = \ell, \mathbf{Y}; \boldsymbol{\Psi})} \\
&= \frac{p(\mathbf{Y} | z_t = k, z_{t-1} = \ell; \boldsymbol{\Psi}) p(z_t = k, z_{t-1} = \ell; \boldsymbol{\Psi})}{\sum_{\ell=1}^{K} \sum_{k=1}^{K} p(\mathbf{Y} | z_t = k, z_{t-1} = \ell; \boldsymbol{\Psi}) p(z_t = k, z_{t-1} = \ell; \boldsymbol{\Psi})} \\
&= \frac{\alpha_{(t-1)\ell} p(\mathbf{y}_t | z_t = k; \boldsymbol{\Psi}) \beta tk A_{\ell k}}{\sum_{\ell=1}^{K} \sum_{k=1}^{K} \alpha_{(t-1)\ell} p(\mathbf{y}_t | z_t = k; \boldsymbol{\Psi}) \beta tk A_{\ell k}}.
\end{aligned} \tag{A.18}$$

A.6 Viterbi algorithm

The Viterbi algorithm (Viterbi, 1967) provides an efficient dynamic programming approach to computing the most likely state sequence $(\hat{z}_1, \ldots, \hat{z}_n)$ that have generated an observation sequence $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$, given a set of HMM parameters:

$$\begin{aligned}
\hat{\mathbf{z}} &= \arg\max_{z_1,\ldots,z_n} p(\mathbf{y}_1, \ldots, \mathbf{y}_n, z_1, \ldots, z_n; \mathbf{\Psi}) \\
&= \arg\max_{z_1,\ldots,z_n} p(z_1)p(\mathbf{y}_1|z_1) \prod_{t=2}^{n} p(z_t|z_{t-1})p(\mathbf{y}_t|z_t) \\
&= \arg\min_{z_1,\ldots,z_n} \big[-\log \pi - \log p(\mathbf{y}_1|z_1) + \sum_{t=2}^{n} - \log p(z_t|z_{t-1}) - \log p(\mathbf{y}_t|z_t) \big]. \quad \text{(A.19)}
\end{aligned}$$

The Viterbi algorithm works on the dynamic programming principle that the minimum cost path to $z_t = k$ is equivalent to the minimum cost path to node z_{t-1} plus the cost of a transition from z_{t-1} to $z_t = k$ (and the cost incurred by observation $\mathbf{y}_t$ given $z_t = k$). The MAP state sequence is then determined by starting at node z_t and reconstructing the optimal path backwards based on the stored calculations. Viterbi decoding reduces the computation cost to $\mathcal{O}(K^2 n)$ operations instead of the brute force $O(K^n)$ operations. The Viterbi algorithm steps are outlined in Algorithm 9.

Algorithm 9 Pseudo code of the Viterbi algorithm for an HMM.

```
Inputs: Observation sequence $(\mathbf{y}_1, \ldots, \mathbf{y}_n)$ and HMM parameters $\mathbf{\Psi}$

1: Initialization: initialize minimum path sum to state $z_1 = k$ for $k = 1, \ldots, K$:
       $S_1(z_1 = k) = -\log \pi_k - \log p(\mathbf{y}_1|z_1 = k)$
2: Recursion: for $t = 2, \ldots, n$ and for $k = 1, \ldots, K$, calculate the minimum path sum
   to state $z_t = k$:
       $S_t(z_t = k) = -\log p(\mathbf{y}_t|z_t = k) + \min_{z_{t-1}} \big[S_{t-1}(z_{t-1}) - \log p(z_t = k|z_{t-1})\big]$
    and let
       $z^*_{t-1}(z_t) = \arg\min_{z_{t-1}} \big[S_{t-1}(z_{t-1}) - \log p(z_t = k|z_{t-1})\big]$
3: Termination: compute $\min_{z_n} S_n(z_n)$ and set $\hat{z}_n = \arg\min_{z_n} S_n(z_n)$
4: State sequence backtracking: iteratively set, for $t = n-1, \ldots, 1$
       $\hat{z}_t = z^*_t(\hat{z}_{t+1})$

Outputs: The most likely state sequence $(\hat{z}_1, \ldots, \hat{z}_n)$.
```

A.7 Concavity proof for the criterion optimized by the IRLS algorithm for the RHLP model

In this section we prove the concavity of the criterion $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$ optimized by the IRLS algorithm (3.20). For this purpose we start by giving the first and the second derivatives of this function.

Since there are $K-1$ parameter vectors $(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_{K-1})$ to be estimated, the gradient vector $\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \mathbf{w}}$ consists of $K-1$ gradient component vectors $\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_k}$ corresponding to the parameter $\boldsymbol{w}_k$ for $k = 1, \ldots, K-1$ and is written as

$$\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \mathbf{w}} = \Big(\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_1}, \ldots, \frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_{K-1}}\Big)^T. \tag{A.20}$$

The gradient of $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$ with respect to each of the $K-1$ parameter vectors $\boldsymbol{w}_k$ is computed as follows:

$$\begin{aligned}
\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_k} &= \sum_{j=1}^{m}\sum_{\ell=1}^{K} \tau_{j\ell}^{(q)} \frac{\partial \log \pi_\ell(t_j; \mathbf{w})}{\partial \boldsymbol{w}_k} \\
&= \sum_{j=1}^{m}\sum_{\ell=1}^{K} \tau_{j\ell}^{(q)} \frac{1}{\pi_\ell(t_j; \mathbf{w})} \frac{\partial \pi_\ell(t_j; \mathbf{w})}{\partial \boldsymbol{w}_k} \\
&= \sum_{j=1}^{m}\sum_{\ell=1}^{K} \tau_{j\ell}^{(q)} \frac{1}{\pi_\ell(t_j; \mathbf{w})} \pi_\ell(t_j; \mathbf{w}) \left(\delta_{k\ell} - \pi_k(t_j; \mathbf{w})\right) \boldsymbol{v}_j \\
&= \sum_{j=1}^{m} \left(\sum_{\ell=1}^{K} \tau_{j\ell}^{(q)} \delta_{k\ell} - \pi_k(t_j; \mathbf{w}) \sum_{\ell=1}^{K} \tau_{j\ell}^{(q)} \right) \boldsymbol{v}_j
\end{aligned} \tag{A.21}$$

where $\delta_{k\ell}$ is the Kronecker symbol ($\delta_{k\ell} = 1$ if $k = \ell$, 0 otherwise). By using the fact that the posterior probabilities sum to one, that is $\sum_{\ell=1}^{K} \tau_{j\ell}^{(q)} = 1$, we then obtain:

$$\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_k} = \sum_{j=1}^{m} \big(\tau_{jk}^{(q)} - \pi_k(t_j; \mathbf{w})\big) \boldsymbol{v}_j \tag{A.22}$$

which is a vector of dimension $(u+1)$. From Equation (A.20), the global gradient vector, that is to say the vector comprising the $K-1$ gradient component vectors $\frac{\partial Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_k}$ $(k = 1, \ldots, K-1)$ where each component gradient vector is given by (A.22), is given by the following $(K-1)\times(u+1)$ dimensional vector:

$$\begin{aligned}
\frac{\partial Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})}{\partial \mathbf{w}} &= \sum_{j=1}^{m}\begin{pmatrix} (\tau_{j1}^{(q)}-\pi_1(t_j;\mathbf{w}))\boldsymbol{v}_j \\ \vdots \\ (\tau_{j,K-1}^{(q)}-\pi_{K-1}(t_j;\mathbf{w}))\boldsymbol{v}_j \end{pmatrix}\\
&= \sum_{j=1}^{m}\begin{pmatrix} \tau_{j1}^{(q)}-\pi_1(t_j;\mathbf{w}) \\ \vdots \\ \tau_{j,K-1}^{(q)}-\pi_{K-1}(t_j;\mathbf{w}) \end{pmatrix}\otimes \boldsymbol{v}_j\\
&= \sum_{j=1}^{m}(\boldsymbol{\tau}_j^{(q)}-\boldsymbol{\pi}(t_j;\mathbf{w}))\otimes\boldsymbol{v}_j \qquad \text{(A.23)}
\end{aligned}$$

where $\otimes$ is the Kronecker product (c.f., Appendix B.3),

$$\boldsymbol{\tau}_j^{(q)} = (\tau_{j1}^{(q)},\ldots,\tau_{j,K-1}^{(q)})^T$$

and

$$\boldsymbol{\pi}(t_j;\mathbf{w}) = \big(\pi_1(t_j;\mathbf{w}),\ldots,\pi_{K-1}(t_j;\mathbf{w})\big)^T.$$

Thus, by evaluating the gradient at $\mathbf{w}=\mathbf{w}^{(l)}$ we obtain:

$$\frac{\partial Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_k}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} = \sum_{j=1}^{m}\big(\tau_{jk}^{(q)}-\pi_k(t_j;\mathbf{w}^{(l)})\big)\boldsymbol{v}_j \qquad \text{(A.24)}$$

and

$$\frac{\partial Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})}{\partial \mathbf{w}}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} = \sum_{j=1}^{m}(\boldsymbol{\tau}_j^{(q)}-\boldsymbol{\pi}(t_j;\mathbf{w}^{(l)}))\otimes\boldsymbol{v}_j. \qquad \text{(A.25)}$$

The Hessian matrix $\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})}{\partial\mathbf{w}\partial\mathbf{w}^T}$ consists of $(K-1)\times(K-1)$ block matrices $\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})}{\partial\boldsymbol{w}_k\partial\boldsymbol{w}_\ell^T}$ for $k,\ell=1,\ldots,K-1$ (Chamroukhi et al., 2009a; Chen et al., 1999), where each block matrix is given by:

$$\begin{aligned}
\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w},\boldsymbol{\theta}^{(q)})}{\partial\boldsymbol{w}_k\partial\boldsymbol{w}_\ell^T} &= \frac{\partial\sum_{j=1}^{m}\left(\tau_{jk}^{(q)}-\pi_k(t_j;\mathbf{w})\right)\boldsymbol{v}_j}{\partial\boldsymbol{w}_\ell^T}\\
&= -\sum_{j=1}^{m}\boldsymbol{v}_j\frac{\partial\pi_k(t_j;\mathbf{w})}{\partial\boldsymbol{w}_\ell^T}\\
&= -\sum_{j=1}^{m}\pi_k(t_j;\mathbf{w})\big(\delta_{k\ell}-\pi_\ell(t_j;\mathbf{w})\big)\boldsymbol{v}_j\boldsymbol{v}_j^T \qquad \text{(A.26)}
\end{aligned}$$

which is a matrix of dimension $(u+1)\times(u+1)$. Thus, by considering all the component matrices for $\ell, k = 1, \ldots, K-1$ in (A.26), the Hessian is therefore given by[1]:

$$\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \mathbf{w} \partial \mathbf{w}^T} = -\sum_{j=1}^{m} \begin{pmatrix} \pi_1(1-\pi_1)\boldsymbol{v}_j\boldsymbol{v}_j^T & -\pi_1\pi_2\boldsymbol{v}_j\boldsymbol{v}_j^T & \ldots & -\pi_1\pi_{K-1}\boldsymbol{v}_j\boldsymbol{v}_j^T \\ -\pi_2\pi_1\boldsymbol{v}_j\boldsymbol{v}_j^T & \pi_2(1-\pi_2)\boldsymbol{v}_j\boldsymbol{v}_j^T & \ldots & -\pi_2\pi_{K-1}\boldsymbol{v}_j\boldsymbol{v}_j^T \\ \vdots & \vdots & \ddots & \vdots \\ -\pi_{K-1}\pi_1\boldsymbol{v}_j\boldsymbol{v}_j^T & -\pi_1\pi_2\boldsymbol{v}_j\boldsymbol{v}_j^T & \ldots & \pi_{K-1}(1-\pi_{K-1})\boldsymbol{v}_j\boldsymbol{v}_j^T \end{pmatrix}$$

$$= -\sum_{j=1}^{m} \begin{pmatrix} \pi_1(1-\pi_1) & -\pi_1\pi_2 & \ldots & -\pi_1\pi_{K-1} \\ -\pi_2\pi_1 & \pi_2(1-\pi_2) & \ldots & -\pi_2\pi_{K-1} \\ \vdots & \vdots & \ddots & \vdots \\ -\pi_{K-1}\pi_1 & -\pi_1\pi_2 & \ldots & \pi_{K-1}(1-\pi_{K-1}) \end{pmatrix} \otimes \boldsymbol{v}_j\boldsymbol{v}_j^T$$

$$= -\sum_{j=1}^{m} \left(\boldsymbol{\Pi}(t_j;\mathbf{w}) - \boldsymbol{\pi}(t_j;\mathbf{w})\boldsymbol{\pi}(t_j;\mathbf{w})^T\right) \otimes \boldsymbol{v}_j\boldsymbol{v}_j^T \tag{A.27}$$

where $\boldsymbol{\Pi}(t_j;\mathbf{w}) = \mathrm{diag}(\pi_1(t_j;\mathbf{w}), \ldots, \pi_{K-1}(t_j;\mathbf{w}))$. Finally, by evaluating the Hessian at $\mathbf{w} = \mathbf{w}^{(l)}$, we obtain:

$$\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \boldsymbol{w}_k \partial \boldsymbol{w}_\ell^T}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} = -\sum_{j=1}^{m} \pi_k(t_j;\mathbf{w}^{(l)})\big(\delta_{k\ell} - \pi_\ell(t_j;\mathbf{w}^{(l)})\big)\boldsymbol{v}_j\boldsymbol{v}_j^T \tag{A.28}$$

and

$$\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \mathbf{w} \partial \mathbf{w}^T}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} = -\sum_{j=1}^{m} \left(\boldsymbol{\Pi}(\mathbf{w}^{(l)}) - \boldsymbol{\pi}(t_j;\mathbf{w}^{(l)})\boldsymbol{\pi}(t_j;\mathbf{w}^{(l)})^T\right) \otimes \boldsymbol{v}_j\boldsymbol{v}_j^T. \tag{A.29}$$

Once we have expressed in this way the Hessian matrix, we show that the criterion $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$ to be maximized by the IRLS algorithm is concave. This can be verified as follows. Let us consider the matrix

$$\mathbf{M}^j = -\big(\boldsymbol{\Pi}(t_j;\mathbf{w}) - \boldsymbol{\pi}(t_j;\mathbf{w})\boldsymbol{\pi}(t_j;\mathbf{w})^T\big).$$

The Hessian matrix (A.29) takes therefore the following form

$$\frac{\partial^2 Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})}{\partial \mathbf{w} \partial \mathbf{w}^T}\Big|_{\mathbf{w}=\mathbf{w}^{(l)}} = \sum_{j=1}^{m} \mathbf{M}^j \otimes \boldsymbol{v}_j\boldsymbol{v}_j^T. \tag{A.30}$$

The matrix $\mathbf{M}^j$ is symmetric since $\forall k, \ell = 1, \ldots, K-1$ we have

$$\mathbf{M}^j_{k\ell} = \begin{cases} -\pi_k(t_j;\mathbf{w}^{(l)})\big(1 - \pi_k(\mathbf{w}^{(l)})\big) & \text{if } k = \ell \\ \pi_k(t_j;\mathbf{w}^{(l)})\pi_\ell(t_j;\mathbf{w}^{(l)}) & \text{if } k \neq \ell \end{cases} \tag{A.31}$$

[1]Here in the expression of the Hessian matrix π_k stands for $\pi_k(t_j;\mathbf{w})$ $(k = 1, \ldots, K-1)$.

which implies $\mathbf{M}^j_{k\ell} = \mathbf{M}^j_{\ell k}$ for $k, \ell = 1, \ldots, K-1$). Additionally, by using the fact that the logistic probabilities for each polynomial component sum to 1, that is $\sum_{\ell=1} \pi_\ell(t_j; \mathbf{w}^{(l)}) = 1$ we also have

$$\begin{aligned}\sum_{\ell \neq k} |\mathbf{M}^j_{k\ell}| = \sum_{\ell \neq k} \pi_k(t_j; \mathbf{w}^{(l)}) \pi_\ell(t_j; \mathbf{w}^{(l)}) &= \pi_k(t_j; \mathbf{w}^{(l)}) \sum_{\ell \neq k} \pi_\ell(t_j; \mathbf{w}^{(l)}) \\ &= \pi_k(t_j; \mathbf{w}^{(l)})(1 - \pi_k(t_j; \mathbf{w}^{(l)})) \\ &\leq |\mathbf{M}^j_{kk}|\end{aligned}$$

which implies that the matrix $\mathbf{M}^j_{kk}$ is diagonally dominant. Since the diagonal entries of $\mathbf{M}^j$ are negative $\mathbf{M}^j_{kk} \leq 0$ (note that $\pi_k(t_j; \mathbf{w}^{(l)}) \geq 0$), $\mathbf{M}^j$ is therefore negative semidefinite. Therefore the Hessian is also negative semidefinite, since for any j, the matrix $\boldsymbol{v}_j \boldsymbol{v}_j^T$ is positive semidefinite and the Kronecker product of a negative semidefinite matrix and a positive semidefinite matrix is also a negative semidefinite matrix. This shows that the criterion $Q_{\mathbf{w}}(\mathbf{w}, \boldsymbol{\theta}^{(q)})$ optimized by the IRLS algorithm is concave (See Appendix B.1 for details on concave (respectively convex) functions).

A.8 The sufficient statistics in the case of the Multivariate Autoregressive Model with Hidden logistic process

For $k = 1, \ldots, K$ we have

$$\begin{aligned}\mathcal{L}_c^{\boldsymbol{\Psi}_k}(\boldsymbol{\Psi}_k; \mathbf{Y}) &= \sum_{t=p+1}^{n} z_{tk} \log \mathcal{N}(\mathbf{y}_t; \mathbf{B}_k^T \mathbf{r}_t, \boldsymbol{\Sigma}_k) \\ &= \frac{-1}{2} \sum_{t=p+1}^{n} z_{tk}^{(q)} \Big[(\mathbf{y}_t - \mathbf{B}_k^T \mathbf{r}_t)^T \boldsymbol{\Sigma}_k^{-1} (\mathbf{y}_t - \mathbf{B}_k^T \mathbf{r}_t) + \log |\boldsymbol{\Sigma}_k| + d \log 2\pi\Big]. \qquad (A.32)\end{aligned}$$

Since for any arbitrary matrix A and a vector $\mathbf{x}$ we have:

$$\mathbf{x}^T A \mathbf{x} = \mathrm{trace}[\mathbf{x}^T A \mathbf{x}] = \mathrm{trace}[\mathbf{x}\mathbf{x}^T A],$$

the distance $D_k = (\mathbf{y}_t - \mathbf{B}_k^T \mathbf{r}_t)^T \boldsymbol{\Sigma}_k^{-1} (\mathbf{y}_t - \mathbf{B}_k^T \mathbf{r}_t)$ is therefore given by:

$$\begin{aligned}D_k &= \mathrm{trace}\big[(\mathbf{y}_t - \mathbf{B}_k^T \mathbf{r}_t)^T \boldsymbol{\Sigma}_k^{-1} (\mathbf{y}_t - \mathbf{B}_k^T \mathbf{r}_t)\big] \\ &= \mathrm{trace}\big[(\mathbf{y}_t - \mathbf{B}_k^T \mathbf{r}_t)(\mathbf{y}_t - \mathbf{B}_k^T \mathbf{r}_t)^T \boldsymbol{\Sigma}_k^{-1}\big] \\ &= \mathrm{trace}\big[\big(\mathbf{y}_t \mathbf{y}_t^T - \mathbf{y}_t \mathbf{r}_t^T \mathbf{B}_k - \mathbf{B}_k^T \mathbf{r}_t \mathbf{y}_t^T + \mathbf{B}_k^T \mathbf{r}_t \mathbf{r}_t^T \mathbf{B}_k\big) \boldsymbol{\Sigma}_k^{-1}\big] \\ &= \mathrm{trace}\big[\mathbf{y}_t \mathbf{y}_t^T \boldsymbol{\Sigma}_k^{-1} - \mathbf{y}_t \mathbf{r}_t^T \mathbf{B}_k \boldsymbol{\Sigma}_k^{-1} - \mathbf{B}_k^T \mathbf{r}_t \mathbf{y}_t^T \boldsymbol{\Sigma}_k^{-1} + \mathbf{B}_k^T \mathbf{r}_t \mathbf{r}_t^T \mathbf{B}_k \boldsymbol{\Sigma}_k^{-1}\big]. \qquad (A.33)\end{aligned}$$

Additionally, for the $d \times d$ matrix $\mathbf{B}_k^T \mathbf{r}_t \mathbf{y}_t^T \boldsymbol{\Sigma}_k^{-1}$ we have

$$\begin{aligned}\mathrm{trace}[\mathbf{B}_k^T \mathbf{r}_t \mathbf{y}_t^T \boldsymbol{\Sigma}_k^{-1}] &= \mathrm{trace}[(\mathbf{B}_k^T \mathbf{r}_t \mathbf{y}_t^T \boldsymbol{\Sigma}_k^{-1})^T] \\ &= \mathrm{trace}[(\boldsymbol{\Sigma}_k^{-1})^T \mathbf{y}_t \mathbf{r}_t^T \mathbf{B}_k] \\ &= \mathrm{trace}[\boldsymbol{\Sigma}_k^{-1} \mathbf{y}_t \mathbf{r}_t^T \mathbf{B}_k] \\ &= \mathrm{trace}[\mathbf{y}_t \mathbf{r}_t^T \mathbf{B}_k \boldsymbol{\Sigma}_k^{-1}] \qquad (A.34)\end{aligned}$$

where in the first step, we used the fact that the trace of a square matrix equals to the trace of its transpose, in the third step we used the fact that $\mathbf{\Sigma}_k^{-1} = (\mathbf{\Sigma}_k^{-1})^T$ since $\mathbf{\Sigma}_k^{-1}$ is the inverse of the positive-definite symmetric covariance matrix $\mathbf{\Sigma}_k$ which is also symmetric, and in the last step we used the fact that since $\mathbf{y}_t\mathbf{r}_t^T\mathbf{B}_k$ is a $d \times d$ matrix (the same dimension of $\mathbf{\Sigma}_k^{-1}$) then we also have: $\text{trace}(\mathbf{\Sigma}_k^{-1}\mathbf{y}_t\mathbf{r}_t^T\mathbf{B}_k) = \text{trace}\,(\mathbf{y}_t\mathbf{r}_t^T\mathbf{B}_k\mathbf{\Sigma}_k^{-1})$. According to the same property ($\text{trace}(FG) = \text{trace}(GF)$), we can also write:

$$\begin{aligned}\text{trace}(\underbrace{\mathbf{B}_k^T\mathbf{r}_t\mathbf{r}_t^T}_{F}\underbrace{\mathbf{B}_k\mathbf{\Sigma}_k^{-1}}_{G}) &= \text{trace}(\underbrace{\mathbf{B}_k\mathbf{\Sigma}_k^{-1}}_{G}\underbrace{\mathbf{B}_k^T\mathbf{r}_t\mathbf{r}_t^T}_{F}) \\ &= \text{trace}(\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\mathbf{\Sigma}_k^{-1}\mathbf{B}_k^T) \qquad \text{(A.35)}\end{aligned}$$

where in the second step we used the fact that $\text{trace}(GF) = \text{trace}((GF)^T) = \text{trace}(F^TG^T)$.

Thus, by using the results (A.34) and (A.35), (A.33) is rewritten as

$$(\mathbf{y}_t - \mathbf{B}_k^T\mathbf{r}_t)^T\mathbf{\Sigma}_k^{-1}(\mathbf{y}_t - \mathbf{B}_k^T\mathbf{r}_t) = \text{trace}\big[\mathbf{y}_t\mathbf{y}_t^T\mathbf{\Sigma}_k^{-1} - 2\mathbf{y}_t\mathbf{r}_t^T\mathbf{B}_k\mathbf{\Sigma}_k^{-1} + \mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\mathbf{\Sigma}_k^{-1}\mathbf{B}_k^T\big] \qquad \text{(A.36)}$$

and we finally obtain:

$$\begin{aligned}\mathcal{L}_c^{\mathbf{\Psi}_k}(\mathbf{\Psi}_k;\mathbf{Y}) = \frac{-1}{2}\text{trace}\Big[\sum_{t=p+1}^{n} z_{tk}\mathbf{y}_t\mathbf{y}_t^T\mathbf{\Sigma}_k^{-1} - 2\sum_{t=p+1}^{n} z_{tk}\mathbf{y}_t\mathbf{r}_t^T\mathbf{B}_k\mathbf{\Sigma}_k^{-1} + \sum_{t=p+1}^{n} z_{tk}\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\mathbf{\Sigma}_k^{-1}\mathbf{B}_k^T\Big] \\ -\frac{1}{2}\sum_{t=p+1}^{n} z_{tk}\log|\mathbf{\Sigma}_k| - \frac{d}{2}\sum_{t=p+1}^{n} z_{tk}\log 2\pi. \qquad \text{(A.37)}\end{aligned}$$

We see then that the complete-data log-likelihood function depends on the data only through the following sufficient statistics:

$$\boldsymbol{T}_k = \sum_{t=p+1}^{n} z_{tk},\ \boldsymbol{T}_{rr,k} = \sum_{t=p+1}^{n} z_{tk}\mathbf{r}_t\mathbf{r}_t^T,\ \boldsymbol{T}_{yr,k} = \sum_{t=p+1}^{n} z_{tk}\mathbf{r}_t\mathbf{y}_t^T,\ \boldsymbol{T}_{yy,k} = \sum_{t=2}^{n} z_{tk}\mathbf{y}_t\mathbf{y}_t^T.$$

A.9 EM algorithm: update formula for the Autoregressive model parameters for the ARHLP model

To maximize (5.25) with respect to the autoregressive model parameters $\mathbf{B}_k$, by collecting together the terms that depend on $\mathbf{B}_k$, we can see that this maximization is equivalent to minimizing the function

$$Q(\mathbf{B}_k) = \text{trace}\Big[-2\sum_{t=p+1}^{n} \tau_{tk}^{(q)}\mathbf{y}_t\mathbf{r}_t^T\mathbf{B}_k\mathbf{\Sigma}_k^{-1} + \sum_{t=p+1}^{n} \tau_{tk}^{(q)}\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\mathbf{\Sigma}_k^{-1}\mathbf{B}_k^T\Big].$$

The derivative of the function $Q(\mathbf{B}_k)$ w.r.t to $\mathbf{B}_k$ is given by:

$$\begin{aligned}
\frac{\partial Q(\mathbf{B}_k)}{\partial \mathbf{B}_k} &= -2\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\frac{\partial \text{trace}\big(\mathbf{y}_t\mathbf{r}_t^T\mathbf{B}_k\boldsymbol{\Sigma}_k^{-1}\big)}{\partial \mathbf{B}_k} + \sum_{t=p+1}^{n}\tau_{tk}^{(q)}\frac{\partial \text{trace}\big(\mathbf{B}_k^T\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\boldsymbol{\Sigma}_k^{-1}\big)}{\partial \mathbf{B}_k}\\
&= -2\sum_{t=p+1}^{n}\tau_{tk}^{(q)}(\mathbf{y}_t\mathbf{r}_t^T)^T(\boldsymbol{\Sigma}_k^{-1})^T + \sum_{t=p+1}^{n}\tau_{tk}^{(q)}\big(\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\boldsymbol{\Sigma}_k^{-1} + (\mathbf{r}_t\mathbf{r}_t^T)^T\mathbf{B}_k(\boldsymbol{\Sigma}_k^{-1})^T\big)\\
&= -2\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{y}_t^T\boldsymbol{\Sigma}_k^{-1} + 2\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\boldsymbol{\Sigma}_k^{-1}\\
&= -2\Big(\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{y}_t^T - \sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\Big)\boldsymbol{\Sigma}_k^{-1} \qquad (A.38)
\end{aligned}$$

where in the first step we used (A.35) and in the second step we used the following results of the derivatives of traces (see for example equations 93 and 106 in the matrix book of Petersen and Pedersen (2008):

$$\frac{\partial \text{trace}\big(\mathbf{F}\mathbf{X}\mathbf{G}\big)}{\partial \mathbf{X}} = \mathbf{F}^T\mathbf{G}^T$$

and

$$\frac{\partial \text{trace}\big(\mathbf{X}^T\mathbf{F}\mathbf{X}\mathbf{G}\big)}{\partial \mathbf{X}} = \mathbf{F}\mathbf{X}\mathbf{G} + \mathbf{F}^T\mathbf{X}\mathbf{G}^T$$

for $\mathbf{F} = \mathbf{r}_t\mathbf{r}_t^T$ and $\mathbf{G} = \boldsymbol{\Sigma}_k^{-1}$. In the second step we used the fact that $(\boldsymbol{\Sigma}_k^{-1})^T = \boldsymbol{\Sigma}_k^{-1}$. Setting (A.38) to zero yields:

$$\Big[\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{r}_t^T\Big]\mathbf{B}_k = \sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{y}_t^T$$

which yields to the following update formula (5.26):

$$\mathbf{B}_k^{(q+1)} = \Big[\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{r}_t^T\Big]^{-1}\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{y}_t^T. \qquad (A.39)$$

A.10 EM algorithm: update formula for the covariance matrix $\boldsymbol{\Sigma}_k$ for the ARHLP model

To maximize (5.25) with respect to the covariance matrix $\boldsymbol{\Sigma}_k$, consider the terms in (5.25) that are function of $\boldsymbol{\Sigma}_k$. By using the fact that $|\boldsymbol{\Sigma}_k| = \frac{1}{|\boldsymbol{\Sigma}_k^{-1}|}$ the function to be maximized can written as:

$$Q(\boldsymbol{\Sigma}_k)=\frac{-1}{2}\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\Big[\text{trace}(\mathbf{y}_t\mathbf{y}_t^T\boldsymbol{\Sigma}_k^{-1})-2\text{trace}(\mathbf{y}_t\mathbf{r}_t^T\mathbf{B}_k\boldsymbol{\Sigma}_k^{-1})+\text{trace}(\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\boldsymbol{\Sigma}_k^{-1}\mathbf{B}_k^T)-\log|\boldsymbol{\Sigma}_k^{-1}|\Big]. \qquad (A.40)$$

If we omit the multiplicative constant in the expression of $Q(\boldsymbol{\Sigma}_k)$, and take its derivative w.r.t $\boldsymbol{\Sigma}_k^{-1}$ we obtain:

$$\begin{aligned}
\frac{\partial Q(\boldsymbol{\Sigma}_k)}{\partial \boldsymbol{\Sigma}_k^{-1}} &= \sum_{t=p+1}^{n} \tau_{tk}^{(q)} \Big[\frac{\partial \text{trace}(\mathbf{y}_t\mathbf{y}_t^T\boldsymbol{\Sigma}_k^{-1})}{\partial \boldsymbol{\Sigma}_k^{-1}} - 2\frac{\partial \text{trace}(\mathbf{y}_t\mathbf{r}_t^T\mathbf{B}_k\boldsymbol{\Sigma}_k^{-1})}{\partial \boldsymbol{\Sigma}_k^{-1}} + \frac{\partial \text{trace}(\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k\boldsymbol{\Sigma}_k^{-1}\mathbf{B}_k^T)}{\partial \boldsymbol{\Sigma}_k^{-1}} \\
&\quad - \frac{\partial \log|\boldsymbol{\Sigma}_k^{-1}|}{\partial \boldsymbol{\Sigma}_k^{-1}}\Big] \\
&= \sum_{t=p+1}^{n} \tau_{tk}^{(q)} \Big[\mathbf{y}_t\mathbf{y}_t^T - 2\mathbf{B}_k^T\mathbf{r}_t\mathbf{y}_t^T + \mathbf{B}_k^T\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k - \boldsymbol{\Sigma}_k\Big] \qquad (A.41)
\end{aligned}$$

where for the two first terms we used the fact that $\frac{\partial \text{trace}(\mathbf{FX})}{\partial \mathbf{X}} = \mathbf{F}^T$ and for the third term we used $\frac{\partial \text{trace}(\mathbf{FXG})}{\partial \mathbf{X}} = \mathbf{F}^T\mathbf{G}^T$ (see for example equations 92 and 93 in Petersen and Pedersen (2008)). For the last term we used the property $\frac{\partial \log|\boldsymbol{\Sigma}_k^{-1}|}{\partial \boldsymbol{\Sigma}_k^{-1}} = ((\boldsymbol{\Sigma}_k^{-1})^{-1})^T = \boldsymbol{\Sigma}_k^T = \boldsymbol{\Sigma}_k$.

Setting (A.41) to zero and replacing $\mathbf{B}_k$ by its update $\mathbf{B}_k^{(q+1)}$ provides the following update formula:

$$\begin{aligned}
\boldsymbol{\Sigma}_k^{(q+1)} &= \frac{\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{y}_t\mathbf{y}_t^T - 2\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{B}_k^{(q+1)T}\mathbf{r}_t\mathbf{y}_t^T + \sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{B}_k^{(q+1)T}\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k^{(q+1)}}{\sum_{t=p+1}^{n}\tau_{tk}^{(q)}} \\
&= \frac{\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{y}_t\mathbf{y}_t^T - \mathbf{B}_k^{(q+1)T}\Big[2\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{y}_t^T - \sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k^{(q+1)}\Big]}{\sum_{t=p+1}^{n}\tau_{tk}^{(q)}} \\
&= \frac{\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{y}_t\mathbf{y}_t^T - \mathbf{B}_k^{(q+1)T}\Big[2\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{y}_t^T - \sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{y}_t^T\Big]}{\sum_{t=p+1}^{n}\tau_{tk}^{(q)}} \\
&= \frac{\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{y}_t\mathbf{y}_t^T - \mathbf{B}_k^{(q+1)T}\sum_{t=p+1}^{n}\tau_{tk}^{(q)}\mathbf{r}_t\mathbf{y}_t^T}{\sum_{t=p+1}^{n}\tau_{tk}^{(q)}} \qquad (A.42)
\end{aligned}$$

where in the third step we used (A.39).

A.11 Recursive updating rule for the parameter $\mathbf{B}_k$ for the ARHLP with a discount factor λ_t

The update of $\mathbf{B}_k$ in basing on the conditional expected sufficient statistics is given by

$$\begin{aligned}
\mathbf{B}_k^{(t+1)} &= \Big[\boldsymbol{S}_{rr,k}^{(t)}\Big]^{-1}\boldsymbol{S}_{yr,k}^{(t)} \\
&= \Big[\boldsymbol{S}_{rr,k}^{(t)}\Big]^{-1}\big((1-\lambda_t)\boldsymbol{S}_{yr,k}^{(t-1)} + \lambda_t\tau_{tk}^{(t-1)}\mathbf{r}_t\mathbf{y}_t^T\big).
\end{aligned}$$

By using the fact that at the time step t we have

$$\mathbf{B}_k^{(t)} = \left[\boldsymbol{S}_{rr,k}^{(t-1)}\right]^{-1} \boldsymbol{S}_{yr,k}^{(t-1)}$$

which implies that

$$\boldsymbol{S}_{yr,k}^{(t-1)} = \boldsymbol{S}_{rr,k}^{(t-1)}\mathbf{B}_k^{(t)},$$

and, additionally from the incremental scheme (5.44) and the expression of $\boldsymbol{S}_{rr,k}^{(t)}$ given in Equation (5.31) we deduce the following relation

$$\boldsymbol{S}_{rr,k}^{(t-1)} = \frac{1}{1-\lambda_t}(\boldsymbol{S}_{rr,k}^{(t)} - \lambda_t\tau_{tk}^{(t-1)}\mathbf{r}_t\mathbf{r}_t^T)$$

we therefore get

$$\boldsymbol{S}_{yr,k}^{(t-1)} = \frac{1}{1-\lambda_t}(\boldsymbol{S}_{rr,k}^{(t)} - \lambda_t\tau_{tk}^{(t-1)}\mathbf{r}_t\mathbf{r}_t^T)\mathbf{B}_k^{(t)}.$$

Finally, the recursive formula is derived as:

$$\begin{aligned}
\mathbf{B}_k^{(t+1)} &= \left[\boldsymbol{S}_{rr,k}^{(t)}\right]^{-1}\left((1-\lambda_t)\boldsymbol{S}_{yr,k}^{(t-1)} + \lambda_t\tau_{tk}^{(t-1)}\mathbf{r}_t\mathbf{y}_t^T\right)\\
&= \left[\boldsymbol{S}_{rr,k}^{(t)}\right]^{-1}\left((1-\lambda_t)\times\frac{1}{1-\lambda_t}(\boldsymbol{S}_{rr,k}^{(t)} - \lambda_t\tau_{tk}^{(t-1)}\mathbf{r}_t\mathbf{r}_t^T)\mathbf{B}_k^{(t)} + \lambda_t\tau_{tk}^{(t-1)}\mathbf{r}_t\mathbf{y}_t^T\right)\\
&= \left[\boldsymbol{S}_{rr,k}^{(t)}\right]^{-1}\boldsymbol{S}_{rr,k}^{(t)}\mathbf{B}_k^{(t)} + \left[\boldsymbol{S}_{rr,k}^{(t)}\right]^{-1}(\lambda_t\tau_{tk}^{(t-1)}\mathbf{r}_t\mathbf{y}_t^T - \lambda_t\tau_{tk}^{(t-1)}\mathbf{r}_t\mathbf{r}_t^T\mathbf{B}_k^{(t)})\\
&= \mathbf{B}_k^{(t)} + \left[\boldsymbol{S}_{rr,k}^{(t)}\right]^{-1}\lambda_t\,\tau_{tk}^{(t-1)}\mathbf{r}_t(\mathbf{y}_t^T - \mathbf{r}_t^T\mathbf{B}_k^{(t)}).
\end{aligned}\tag{A.43}$$

A.12 Derivatives of the complete-data log-likelihood for the AR-NH-HMM model w.r.t the parameters of the hidden process

The derivatives of the function (5.55) we attempt to maximize w.r.t $\mathbf{w}_\ell$, using the IRLS algorithm, for $\ell = 1,\ldots,K$ where $\mathbf{w}_\ell = (\boldsymbol{w}_1^\ell,\ldots,\boldsymbol{w}_K^\ell)$ are given as follows. In (5.55) we have $\mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{w}_\ell) = \frac{\exp(\boldsymbol{w}_k^{\ell\,T}\mathbf{y}_{t-1})}{\sum_{k=1}^K\exp(\boldsymbol{w}_k^{\ell\,T}\mathbf{y}_{t-1})}$. The first derivative of $\mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{z})$ is therefore given by:

$$\begin{aligned}
\frac{\partial \mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{z})}{\partial \boldsymbol{w}_h^{\ell}} &= \sum_{k=1}^{K}\sum_{t=2}^{n} z_{t-1,\ell} z_{tk} \frac{\partial \log \mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{w}_\ell)}{\partial \boldsymbol{w}_h^{\ell}}\\
&= \sum_{k=1}^{K}\sum_{t=2}^{n} z_{t-1,\ell} z_{tk} \frac{1}{\mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{w}_\ell)} \frac{\partial \mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{w}_\ell)}{\partial \boldsymbol{w}_h^{\ell}}\\
&= \sum_{k=1}^{K}\sum_{t=2}^{n} z_{t-1,\ell} z_{tk} \frac{1}{\mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{w}_\ell)} \mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{w}_\ell)\left(\delta_{kh} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1};\mathbf{w}_\ell)\right)\mathbf{y}_{t-1}\\
&= \sum_{t=2}^{n}\left(\sum_{k=1}^{K} z_{t-1,\ell} z_{tk}\delta_{kh} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1};\mathbf{w}_\ell)\sum_{k=1}^{K}\xi_{t\ell k}^{(q)}\right)\mathbf{y}_{t-1}\\
&= \sum_{t=2}^{n}\left(z_{t-1,\ell} z_{th} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1};\mathbf{w}_\ell)\sum_{k=1}^{K} z_{t-1,\ell} z_{tk}\right)\mathbf{y}_{t-1}\\
&= \sum_{t=2}^{n}\left(z_{t-1,\ell} z_{th} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1};\mathbf{w}_\ell) z_{t-1,\ell}\right)\mathbf{y}_{t-1} \qquad \text{(A.44)}
\end{aligned}$$

where in the third step we used the derivation result in (A.2) and in the last step we used the fact that $\sum_{k=1}^{K} z_{t-1,\ell} z_{tk} = z_{t-1,\ell}\sum_{k=1}^{K} z_{tk} = z_{t-1,\ell}$.

The Hessian matrix is composed of $(K-1)\times(K-1)$ block matrices where each block matrix is of dimension $(d\times d)$ and is given by:

$$\begin{aligned}
\frac{\partial^2 \mathcal{L}_c(\boldsymbol{\Psi};\mathbf{Y},\mathbf{z})}{\partial \boldsymbol{w}_h^{\ell}\partial \boldsymbol{w}_k^{\ell^T}} &= \sum_{t=2}^{n} \frac{\partial\left(z_{t-1,\ell} z_{th} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1};\mathbf{w}_\ell) z_{t-1,\ell}\right)\mathbf{y}_{t-1}}{\partial \boldsymbol{w}_k^{\ell^T}}\\
&= -\sum_{t=2}^{n} z_{t-1,\ell}\mathbf{y}_{t-1}\frac{\partial \mathbf{A}_{\ell h}(\mathbf{y}_{t-1};\mathbf{w}_\ell)}{\partial \boldsymbol{w}_k^{T}}\\
&= -\sum_{t=2}^{n} z_{t-1,\ell}\mathbf{y}_{t-1}\mathbf{A}_{\ell h}(\mathbf{y}_{t-1};\mathbf{w}_\ell)\left(\delta_{hk} - \mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{w}_\ell)\right)\mathbf{y}_{t-1}^{T}\\
&= -\sum_{t=2}^{n} z_{t-1,\ell}\mathbf{A}_{\ell h}(\mathbf{y}_{t-1};\mathbf{w}_\ell)\left(\delta_{hk} - \mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{w}_\ell)\right)\mathbf{y}_{t-1}\mathbf{y}_{t-1}^{T}. \qquad \text{(A.45)}
\end{aligned}$$

A.13 Derivatives of the Q-function w.r.t the parameters of the hidden process for the AR-NH-HMM model

Here we give the derivatives of the function (5.74) given by

$$Q_{\boldsymbol{w}}(\mathbf{w}_\ell,\boldsymbol{\Psi}^{(q)}) = \sum_{k=1}^{K}\sum_{t=2}^{n}\xi_{t\ell k}^{(q)}\log \mathbf{A}_{\ell k}(\mathbf{y}_{t-1};\mathbf{w}_\ell) \qquad \text{(A.46)}$$

we attempt to maximize w.r.t $\mathbf{w}_\ell$ for $\ell = 1, \ldots, K$ using the IRLS algorithm where $\mathbf{w}_\ell = (\boldsymbol{w}_1^\ell, \ldots, \boldsymbol{w}_K^\ell)$ and $\mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell) = \frac{\exp(\boldsymbol{w}_k^{\ell\,T} \mathbf{y}_{t-1})}{\sum_{k=1}^K \exp(\boldsymbol{w}_k^{\ell\,T} \mathbf{y}_{t-1})}$. The first derivative of $Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \mathbf{\Psi}^{(q)})$ is given by:

$$\begin{aligned}
\frac{\partial Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \mathbf{\Psi}^{(q)})}{\partial \boldsymbol{w}_h^\ell} &= \sum_{k=1}^K \sum_{t=2}^n \xi_{t\ell k}^{(q)} \frac{\partial \log \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell)}{\partial \boldsymbol{w}_h^\ell} \\
&= \sum_{k=1}^K \sum_{t=2}^n \xi_{t\ell k}^{(q)} \frac{1}{\mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell)} \frac{\partial \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell)}{\partial \boldsymbol{w}_h^\ell} \\
&= \sum_{k=1}^K \sum_{t=2}^n \xi_{t\ell k}^{(q)} \frac{1}{\mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell)} \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell) \left(\delta_{kh} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1}; \mathbf{w}_\ell)\right) \mathbf{y}_{t-1} \\
&= \sum_{t=2}^n \left(\sum_{k=1}^K \xi_{t\ell k}^{(q)} \delta_{kh} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1}; \mathbf{w}_\ell) \sum_{k=1}^K \xi_{t\ell k}^{(q)} \right) \mathbf{y}_{t-1} \\
&= \sum_{t=2}^n \left(\xi_{t\ell h}^{(q)} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1}; \mathbf{w}_\ell) \sum_{k=1}^K \xi_{t\ell k}^{(q)} \right) \mathbf{y}_{t-1} \\
&= \sum_{t=2}^n \left(\xi_{t\ell h}^{(q)} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1}; \mathbf{w}_\ell) \tau_{t-1,\ell}^{(q)} \right) \mathbf{y}_{t-1} \qquad \text{(A.47)}
\end{aligned}$$

where in the third step we used the derivation result in (A.2) and in the last step we used the fact that $\sum_{k=1}^K \xi_{t\ell k} = \sum_{k=1}^K p(z_t = k, z_{t-1} = \ell | \mathbf{Y}, \mathbf{\Psi}) = p(z_{t-1} = \ell | \mathbf{Y}, \mathbf{\Psi}) = \tau_{t-1,\ell}$.

The Hessian matrix is composed of $(K-1) \times (K-1)$ block matrices where each block matrix is of dimension $(d \times d)$ and is given by:

$$\begin{aligned}
\frac{\partial^2 \partial Q_{\boldsymbol{w}}(\mathbf{w}_\ell, \mathbf{\Psi}^{(q)})}{\partial \boldsymbol{w}_h^\ell \partial \boldsymbol{w}_k^{\ell\,T}} &= \sum_{t=2}^n \frac{\partial \left(\xi_{t\ell h}^{(q)} - \mathbf{A}_{\ell h}(\mathbf{y}_{t-1}; \mathbf{w}_\ell) \tau_{t-1,\ell}^{(q)} \right) \mathbf{y}_{t-1}}{\partial \boldsymbol{w}_k^{\ell\,T}} \\
&= -\sum_{t=2}^n \tau_{t-1,\ell}^{(q)} \mathbf{y}_{t-1} \frac{\partial \mathbf{A}_{\ell h}(\mathbf{y}_{t-1}; \mathbf{w}_\ell)}{\partial \boldsymbol{w}_k^T} \\
&= -\sum_{t=2}^n \tau_{t-1,\ell}^{(q)} \mathbf{y}_{t-1} \mathbf{A}_{\ell h}(\mathbf{y}_{t-1}; \mathbf{w}_\ell) \left(\delta_{hk} - \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell)\right) \mathbf{y}_{t-1}^T \\
&= -\sum_{t=2}^n \tau_{t-1,\ell}^{(q)} \mathbf{A}_{\ell h}(\mathbf{y}_{t-1}; \mathbf{w}_\ell) \left(\delta_{hk} - \mathbf{A}_{\ell k}(\mathbf{y}_{t-1}; \mathbf{w}_\ell)\right) \mathbf{y}_{t-1} \mathbf{y}_{t-1}^T . \text{(A.48)}
\end{aligned}$$

Appendix B

Appendix B

B.1 Convex function

Let f be a real-valued function whose domain is $\mathcal{X}$. f is a convex function if for any two points x_1 and x_2 in its domain $\mathcal{X}$ and any $\lambda \in [0, 1]$,

$$f(\lambda x_1 + (1 - \lambda)x_2) \leq f(x_1) + (1 - \lambda)f(x_2). \tag{B.1}$$

If f is twice differentiable everywhere in $\mathcal{X}$, f is a convex function if $\frac{\partial^2 f}{\partial^2 x} \geq 0$ for all $x \in \mathcal{X}$. In the case of f taking vector-valued inputs, this is generalized to the condition that its Hessian $\frac{\partial^2 f}{\partial \mathbf{x} \partial \mathbf{x}^T}$ is positive semi-definite ($\frac{\partial^2 f}{\partial \mathbf{x} \partial \mathbf{x}^T} \geq 0$).

B.2 Jensen's inequality

Jensen's inequality is stated as follows. Let f be a convex function whose domain is $\mathcal{X}$, and let X be a random variable. Then:

$$\mathbb{E}[f(X)] \geq f(\mathbb{E}[X])$$

Remark: Jensen's inequality also holds for a concave function f, but with the direction of the inequality reversed

$$\mathbb{E}[f(X)] \leq f(\mathbb{E}[X])$$

since the function f is concave if and only if $-f$ is convex (i.e., the Hessian $\frac{\partial^2 f}{\partial \mathbf{x} \partial \mathbf{x}^T}$ is negative semi-definite, written $\frac{\partial^2 f}{\partial \mathbf{x} \partial \mathbf{x}^T} \leq 0$).

B.3 The Kronecker product

If A is a $m \times m$ matrix and B is a $n \times n$ matrix, then the Kronecker product of A and B is the $(m \times m) \times (n \times n)$ matrix

$$A \otimes B = \begin{pmatrix} a_{11}B & \dots & a_{1m}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \dots & a_{mm}B \end{pmatrix} \tag{B.2}$$

For the properties of the Kronecker product, see (Alan, 2005, Chapter 13).

B.4 Hadamard product

Let A and B be $m \times n$ matrices. The Hadamard product of A and B is defined by $(A \circ B)_{ij} = A_{ij}B_{ij}$ for all $1 \leq i \leq m, 1 \leq j \leq n$. The Hadamard product is simply entrywise multiplication.

B.5 Construction of B-splines basis functions

Given the sequence of knots $\zeta_0 < \zeta_1 < \ldots < \zeta_{K+1}$ (ζ_0 and ζ_{K+1} are the two bounds of t), let us define the augmented knot sequence τ such that

- $\tau_1 \leq \tau_2 \ldots \leq \tau_M \leq \zeta_0$;
- $\tau_{M+k} = \zeta_k,\ k = 1, \ldots, K$;
- $\zeta_{K+1} \leq \tau_{K+M+1} \leq \tau_{K+M+2} \ldots \leq \tau_{K+2M}$.

The actual values of these additional knots beyond the boundary are arbitrary, and a common choice is to make them all the same and equal to ζ_0 and ζ_{K+1} respectively.

Let us denote by $B_{j,M}(t)$ the jth B-spline basis function of order M for the knot-sequence

$$\tau_1 \leq \tau_2 \ldots \leq \tau_M \leq \zeta_0 < \zeta_1 < \ldots < \zeta_K < \zeta_{K+1} \leq \tau_{K+M+1} \leq \tau_{K+M+2} \ldots \leq \tau_{K+2M}.$$

These basis functions are defined recursively as follows:

- $B_{j,1}(t_i) = \mathbb{1}_{[\tau_j, \tau_{j+1}]},\ \forall j = 1, \ldots, K + 2M - 1$;
- $B_{j,M}(t_i) = \frac{t_i - \tau_j}{\tau_{j+M-1} - \tau_j} B_{j,M-1}(t_i) + \frac{\tau_{j+M} - t_i}{\tau_{j+M} - \tau_{j+1}} B_{j+1,M-1}(t_i),\ \forall j = 1, \ldots, K + M.$

For the B-spline regression model, the ith row $\mathbf{t}_i$ of the $n \times (M + K)$ regression matrix $\mathbf{X}$ is then constructed as follows:

$$\mathbf{t}_i = [B_{1,M}(t_i),\ B_{2,M}(t_i), \ldots,\ B_{M+K,M}(t_i)].$$

B.6 Construction of natural cubic splines basis functions

The natural cubic splines is a spline regression model with constraints that consist in considering that the spline function is linear before the first internal knot (in $[\zeta_0, \zeta_1]$) and after the last internal knot (in $[\zeta_K, \zeta_{K+1}]$)). These constraints imply null second and third derivatives of the spline function in these regions. Formally one obtains $\beta_1 = 0,\ \beta_2 = 0, \sum_{k=1}^{K} \beta_{k+2} = 0,\ \sum_{k=1}^{K} \zeta_k \beta_{k+2} = 0$). The natural cubic spline function is given by:

$$f(t_i) = \sum_{j=1}^{K+2} \beta_j h_j(t_i) \tag{B.3}$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{K+2})^T$ is the vector of spline coefficients and the truncated power basis $h_j(t_i)$ are defined as follows:

- $h_j(t_i) = t_i^{j-1},\ j = 1, 2$
- $h_{k+2}(t_i) = d_k(t_i) - d_{K-1}(t_i),\ k = 1, \ldots, K-2,$

with $d_k(t_i) = \frac{(t_i-\zeta_k)^3_+ - (t_i-\zeta_K)^3_+}{\zeta_K - \zeta_k}$. Thus, each row of the regression matrix $\mathbf{X}$, in the case of natural cubic splines is constructed as follows:

$$\mathbf{t}_i = [1,\ t_i,\ d_k(t_i) - d_{K-1}(t_i)].$$

Bibliography

H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

J. L. Alan. *Matrix Analysis for Scientists and Engineers.* SIAM: Society for Industrial and Applied Mathematics, Philadelphia, PA, 2005.

J. Alon, S. Sclaroff, G. Kollios, and V. Pavlovic. Discovering Clusters in Motion Time-Series Data. pages 375–381, Los Alamitos, CA, USA, 2003.

S. Aseervatham and Y. Bennani. Semi-structured document categorization with a semantic kernel. *Pattern Recognition*, 42(9):2067–2076, 2009.

F. R. Bach and M. I. Jordan. Learning Spectral Clustering, With Application To Speech Separation. *Journal of Machine Learning Research*, 7:1963–2001, 2006.

P. Baldi and Y. Chauvin. Smooth on-line learning algorithms for hidden Markov models. *Neural Computation*, 6(2):307–318, 1994.

J. D. Banfield and A. E. Raftery. Model-Based Gaussian and Non-Gaussian Clustering. *Biometrics*, 49(3):803–821, 1993.

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

R. Bellman. On the approximation of curves by line segments using dynamic programming. *Communications of the Association for Computing Machinery (CACM)*, 4(6): 284, 1961.

D. Benboudjema and W. Pieczynski. Unsupervised Statistical Segmentation of Nonstationary Images Using Triplet Markov Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(8):1367–1378, 2007.

Y. Bengio and P. Frasconi. An input output HMM architecture. In *Advances in Neural Information Processing Systems*, volume 7, pages 427–434, 1995.

Y. Bengio and P. Frasconi. Input Output HMM's for Sequences Processing. *IEEE Transactions on Neural Networks*, 7(5), 1996.

C. Biernacki, G. Celeux, and G. Govaert. An improvement of the NEC criterion for assessing the number of clusters in a mixture model. *Pattern Recognition Letters*, 20 (3):267–272, 1999.

C. Biernacki, G. Celeux, and G Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):719–725, 2000.

C. Biernacki, G. Celeux, and G. Govaert. Choosing Starting Values for the EM Algorithm for Getting the Highest Likelihood in Multivariate Gaussian Mixture Models. *Computational Statistics and Data Analysis*, 41:561–575, 2003.

C. M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, USA, 1995.

C. M. Bishop. *Pattern recognition and machine learning.* Springer Verlag, U K, 2006.

K. Bleakley and J.-P. Vert. Joint segmentation of many aCGH profiles using fast group LARS. Technical report, HAL-00422430, 2009.

A. Bordes. *New Algorithms for Large-Scale Support Vector Machines.* PhD thesis, Université Pierre et Marie Curie, Computer Science Laboratory of Paris 6 (LIP6), February 2010.

L. Bottou. Online Algorithms and Stochastic Approximations. In David Saad, editor, *Online Learning and Neural Networks.* Cambridge University Press, Cambridge, UK, 1998.

L. Bottou. Stochastic Learning. In O. Bousquet and U. Von Luxburg, editors, *Advanced Lectures on Machine Learning*, Lecture Notes in Artificial Intelligence, LNAI 3176, pages 146–168. Springer Verlag, Berlin, 2004.

L. Bottou and Y. Bengio. Convergence Properties of the K-Means Algorithms. In *Advances in Neural Information Processing Systems 7*, pages 585–592. MIT Press, 1995.

S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004. URL http://www.stanford.edu/~boyd/cvxbook/.

H. Bozdogan. Determining the Number of Component Clusters in the Standard Multivariate Normal Mixture Model Using Model-Selection Criteria. Technical report, Quantitative Methods Department, University of Illinois at Chicago, June 1983.

V. L. Brailovsky and Y. Kempner. Application of piecewise regression to detecting internal structure of signal. *Pattern recognition*, 25(11):1361–1370, 1992.

L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification And Regression Trees.* Wadsworth, New York, 1984.

H. Caillol, W. Pieczynski, and A. Hillion. Estimation of fuzzy Gaussian mixture and unsupervised statistical image segmentation. *IEEE Transactions on Image Processing*, 6(3):425–440, 1997.

O. Cappé. Online EM Algorithm for Hidden Markov Models. *preprint*, August 2009. URL http://arxiv.org/abs/0908.2359.

O. Cappé and E. Moulines. On-line expectation-maximization algorithm for latent data models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(3):593–613, June 2009.

O. Cappé, E. Moulines, and T. Rydén. *Inference in Hidden Markov Models*. Springer Series in Statistics. Springer, July 2005.

A. X. Carvalho and M. A. Tanner. Modeling nonlinearities with mixtures-of-experts of time series models. *International journal of mathematics and mathematical sciences*, 2006(6):1–22, 2006.

A. X. Carvalho and M. A. Tanner. Modelling nonlinear count time series with local mixtures of Poisson autoregressions. *Computational Statistics and Data Analysis*, 51 (11):5266–5294, 2007.

G. Celeux and J. Diebolt. The SEM algorithm a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. *Computational Statistics Quarterly*, 2(1):73–82, 1985.

G. Celeux and J. Diebolt. A stochastic approximation type EM algorithm for the mixture problem. Technical Report RR-1383, The French National Institute for Research in Computer Science and Control (INRIA), 1991.

G. Celeux and J. Diebolt. A Stochastic Approximation Type EM Algorithm for the Mixture Problem. *Stochastics and Stochastics Reports*, 41:119–134, 1992.

G. Celeux and G. Govaert. A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics and Data Analysis*, 14:315–332, 1992.

G. Celeux and G. Govaert. Comparison of the Mixture and the Classification Maximum likelihood in Cluster Analysis. *Journal of Statistical Computation and Simulation*, 47:127–146, 1993.

G. Celeux and G. Govaert. Gaussian parsimonious clustering models. *Pattern Recognition*, 28(5):781–793, 1995.

G. Celeux and G. Soromenho. An entropy criterion for assessing the number of clusters in a mixture model. Technical Report RR-1874, The French National Institute for Research in Computer Science and Control (INRIA), 1993.

G. Celeux and G. Soromenho. An entropy criterion for assessing the number of clusters in a mixture model. *Journal of Classification*, 13(2):195–212, September 1996.

G. Celeux, D. Chauveau, and J. Diebolt. On Stochastic Versions of the EM Algorithm. Technical Report RR-2514, The French National Institute for Research in Computer Science and Control (INRIA), 1995.

G. Celeux, D. Chauveau, and J. Diebolt. Stochastic versions of the EM algorithm: an experimental study in the mixture case. *Journal of statistical computation and simulation*, 55(4):287–314, 1996.

G. Celeux, J.C. Nascimento, and J.S. Marques. Learning switching dynamic models for objects tracking. *PR*, 37(9):1841–1853, September 2004.

F. Chamroukhi, A. Samé, G. Govaert, and P. Aknin. A regression model with a hidden logistic process for feature extraction from time series. In *International Joint Conference on Neural Networks (IJCNN)*, 2009a.

F. Chamroukhi, A. Samé, G. Govaert, and P. Aknin. A regression model with a hidden logistic process for signal parameterization. *Proceedings of XVIIth European Symposium on Artificial Neural Networks ESANN*, pages 503–508, 2009b.

F. Chamroukhi, A. Samé, G. Govaert, and P. Aknin. Time series modeling by a regression approach based on a latent process. *Neural Networks*, 22(5-6):593–602, 2009c.

F. Chamroukhi, A. Samé, G. Govaert, and P. Aknin. A hidden process regression model for functional data description. Application to curve discrimination. *Neurocomputing*, 73(7-9):1210–1221, March 2010.

C. Chemudugunta, P. Smyth, and M. Steyvers. Modeling General and Specific Aspects of Documents with a Probabilistic Topic Model. In *Advances in Neural Information Processing Systems*, 2006.

K. Chen, L. Xu, and H. Chi. Improved learning algorithms for mixture of experts in multiclass classification. *Neural Networks*, 12(9):1229–1252, 1999.

E. Côme. *Apprentissage de modèles génératifs pour le diagnostic de systèmes complexes avec labellisation douce et contraintes spatiales.* Thèse de doctorat, Université de Technologie de Compiègne, 2009.

M. V. Dang. *Classification de Données Spatiales : Modèles Probabilistes et Critères de Partitionnement.* PhD thesis, Université de Technologie de Compiègne, 1998.

A. Debiolles. *Diagnostic de systèmes complexes base de modèle interne, reconnaissance des formes et fusion d'informations. Application au diagnostic des Cricuits de Voie ferroviaires.* Thèse de doctorat, Universitè de Technologie de Compiègne, 2007.

C. Deboor. *A Practical Guide to Splines.* Springer-Verlag, 1978.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of The Royal Statistical Society, B*, 39(1):1–38, 1977.

L. Derrode, S. Benyoussef and W. Pieczynski. Contextual estimation of hidden Markov chains with application to image segmentation. In *ICASSP*, pages 15–19, Toulouse, May 2006.

F.X. Diebold, J.-H. Lee, and G. Weinbach. Regime Switching with Time-Varying Transition Probabilities. *Nonstationary Time Series Analysis and Cointegration. (Advanced Texts in Econometrics)*, pages 283–302, 1994.

R. Douc, E. Moulines, and T. Ryden. Asymptotic properties of the maximum likelihood estimator in autoregressive models with Markov regime. *Annals of Statistics*, 32(5): 2254–2304, 2004.

B. Dubuisson. *Diagnostic et Reconnaissance des formes*. Hermès, 1990.

G. Ferrari-Trecate and M. Muselli. A new learning method for piecewise linear regression. In *International Conference on Artificial Neural Networks (ICANN)*, pages 28–30, 2002.

G. Ferrari-Trecate, M. Muselli, D. Liberati, and M. Morari. A clustering technique for the identification of piecewise affine and hybrid systems. *Automatica*, 39:205–217, 2002.

W. D. Fisher. On grouping for maximum homogeneity. *Journal of the American Statistical Association*, 53:789–798, 1958.

G. D. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

C. Fraley and A. E. Raftery. How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis. *The Computer Journal*, 41(8):578–588, August 1998.

C. Fraley and A. E. Raftery. Model-Based Clustering, Discriminant Analysis, and Density Estimation. *Journal of the American Statistical Association*, 97:611–631, 2002.

M. Fridman. Hidden Markov Model Regression. Technical report, Institute of mathematics, University of Minnesota, 1993.

S. Frühwirth-Schnatter. *Finite Mixture and Markov Switching Models (Springer Series in Statistics)*. Springer Verlag, New York, 2006.

S. J. Gaffney. *Probabilistic Curve-Aligned Clustering and Prediction with Regression Mixture Models*. PhD thesis, Department of Computer Science, University of California, Irvine., 2004.

S. J. Gaffney and P. Smyth. Joint probabilistic curve clustering and alignment. In *In Advances in Neural Information Processing Systems (NIPS)*, 2004.

J-L Gauvain and C-H Lee. Bayesian learning for hidden Markov model with Gaussian mixture state observation densities. *Speech Communication*, 11(2-3):205–213, 1992.

J-L Gauvain and C-H Lee. Maximum a Posteriori Estimation for Multivariate Gaussian Mixture Observations of Markov Chains. *IEEE Transactions on Speech and Audio Processing*, 2(2):291–298, April 1994.

P. Green. Iteratively Reweighted Least Squares for Maximum Likelihood Estimation, and some robust and resistant alternatives. *Journal of The Royal Statistical Society, B*, 46(2):149–192, 1984.

J. Gui and H. Li. Mixture functional discriminant analysis for gene function classification based on time course gene expression data. In *Proc. Joint Stat. Meeting (Biometric Section)*, 2003.

D. Hand, F. Daly, A. D. Lunn, K. J. McConway, and E. Ostrowski. *A Handbook of Small Data Sets*. Chapman and Hall., 1994.

L. Harrison, W. D. Penny, and K. Friston. Multivariate autoregressive modeling of fMRI time series. *Neuroimage*, 19(4):1477–1491, August 2003.

T. Hastie and R. Tibshirani. Discriminant Analysis by Gaussian Mixtures. *Journal of the Royal Statistical Society, B*, 58:155–176, 1996.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning, Second Edition: Data Mining, Inference, and Prediction*. Springer Series in Statistics. Springer, second edition edition, January 2010.

G. Hébrail, B. Hugueney, Y. Lechevallier, and F. Rossi. Exploratory analysis of functional data via clustering and optimal segmentation. *Neurocomputing*, 73(7-9):1125–1141, March 2010.

D. W. Hosmer and S. Lemeshow. *Applied logistic regression (Wiley Series in probability and statistics)*. Wiley-Interscience Publication, September 2000.

J. P. Hughes, P. Guttorp, and S. P. Charles. A non-homogeneous hidden Markov model for precipitation occurrence. *Applied Statistics*, 48:15–30, 1999.

B. Hugueney, G. Hébrail, Y. Lechevallier, and F. Rossi. Simultaneous Clustering and Segmentation for Functional Data. In *Proceedings of XVIIth European Symposium on Artificial Neural Networks (ESANN 2009)*, pages 281–286, Bruges, Belgium, April 2009.

R. Isermann. Process fault detection based on modeling and estimation methods. *Automatica*, 20(4):387–404, 1984.

R. Isermann. Model-based fault detection and diagnosis: status and applications. In *In Proceedings of the 16th IFAC Symposium on Automatic Control in Aerospace, St*, pages 71–85, 2004.

R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive Mixtures of Local Experts. *Neural Computation*, 3(1):79–87, 1991.

G. M. James and T. J. Hastie. Functional Linear Discriminant Analysis for Irregularly Sampled Curves. *Journal of the Royal Statistical Society Series B*, 63:533–550, 2001.

G. M. James and C. Sugar. Clustering for Sparsely Sampled Functional Data. *Journal of the American Statistical Association*, 98(462), 2003.

T. Jebara. *Discriminative, Generative and Imitative learning*. Phd thesis, Media Laboratory, MIT, 2001.

T. Jebara. *Machine Learning: Discriminative and Generative (Kluwer International Series in Engineering and Computer Science)*. Kluwer Academic Publishers, Norwell, MA, USA, 2003.

T. Jebara, Y. Song, and K. Thadani. Spectral Clustering and Embedding with Hidden Markov Models. In *Machine Learning: ECML 2007*, pages 164–175, 2007.

J. L. W. V. Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, 30(1):175–193, 1906.

T. Joachims. *Learning to Classify Text Using Support Vector Machines – Methods, Theory, and Algorithms*. Kluwer/Springer, 2002.

M. I. Jordan and R. A. Jacobs. Hierarchical Mixtures of Experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.

B.-H. Juang and L. R. Rabiner. Mixture autoregressive hidden Markov models for speech signals. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 33 (6):1404–1413, December 1985.

B.-H. Juang, S. E. Levinson, and M. M. Sondhi. Maximum likelihood estimation for mixture multivariate stochastic observations of Markov chains. In *IEEE International Symposium on Information Theory*, Brighton, England, June 1985.

B.-H. Juang, S. E. Levinson, and M. M. Sondhi. Maximum likelihood estimation for multivariate mixture observations of Markov chains. *IEEE Transactions on Information Theory*, 32(2):307–309, March 1986.

P. Kenny, M. Lennig, and Mermelstein P. A linear predictive HMM for vector-valued observations with applications to speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 38(2):220–225,, 1990.

J. J. Kivinen, E. B. Sudderth, and M. I. Jordan. Image Denoising with Nonparametric Hidden Markov Trees. In *ICIP (3)*, pages 121–124, 2007.

T. Kohonen. *Self-Organizing Maps*. Information Sciences. Springer, third edition edition, 2001.

S. Kong and B. Kosko. Differential competitive learning for centroid estimation and phoneme recognition. *IEEE Transactions on Neural Networks*, 2(1):118–124, 1991.

B. Krishnapuram, L. Carin, M.A.T. Figueiredo, and A.J. Hartemink. Sparse multinomial logistic regression: fast algorithms and generalization bounds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):957–968, 2005.

K. Lange. A gradient algorithm locally equivalent to the EM algorithm. *Journal of the Royal Statistical Society, B*, 57:425–437., 1995.

Y. Lechevalier. Optimal clustering on ordered set. *Technical report, French National Institute for Research in Computer Science and Control (INRIA)*, 1990.

C. Liu and D. B. Rubin. The ECME algorithm: A simple extension of EM and ECM with faster monotone convergence. *Biometrika*, 81(4):633–648, December 1994. doi: 10.1093/biomet/81.4.633.

X. Liu and M.C.K. Yang. Simultaneous curve registration and clustering for functional data. *Computational Statistics and Data Analysis*, 53(4):1361–1376, 2009.

J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

V. Makarenkov and P Legendre. Une méthode d'analyse canonique non-linéaire et son application à des données biologiques. *Mathématiques, informatique et sciences humaines*, 1999.

V. E. McGee and W. T. Carleton. Piecewise regression. *Journal of the American Statistical Association*, 65:1109–1124, 1970.

G. J. McLachlan and T. Krishnan. *The EM algorithm and extensions.* New York: Wiley, 1997.

G. J. McLachlan and D. Peel. *Finite mixture models.* New York: Wiley, 2000.

G.J. McLachlan and K.E. Basford. *Mixture Models: Inference and Applications to Clustering.* Marcel Dekker, New York, 1988.

M. Meila and M. I. Jordan. Learning fine motion by Markov mixtures of experts. In *Advances in Neural Information Processing Systems 8*, pages 1003–1009. MIT Press, 1996.

X. L. Meng and D. B. Rubin. Maximum likelihood estimation via the ECM algorithm: A general framework. *Biometrika*, 80(2):267–278, 1993.

T. P. Minka. Algorithms for maximum-likelihood logistic regression. Technical Report 758, Carnegie Mellon University, 2001.

T. M. Mitchell. *Machine Learning.* McGraw-Hill, New York, 1997.

G. Mongillo and S. Deneve. Online learning with hidden markov models. *Neural computation*, 20(7):1706–1716, July 2008.

F. Muri. *Comparaison d'algorithmes d'identification de chaînes de Markov cachées et application à la détection de régions homogènes dans les séquences ADN*. PhD thesis, Université Paris Descartes, Paris V, 1997.

K. P. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning*. PhD thesis, UC Berkeley, Computer Science Division, 2002.

R. Neal and G. E. Hinton. *A view of the EM algorithm that justifies incremental, sparse, and other variants*, pages 355–368. Dordrecht: Kluwer Academic Publishers, 1998.

S-K. Ng, G. J. McLachlan, and A. H. Lee. An incremental EM-based learning approach for on-line prediction of hospital resource utilization. *Artificial Intelligence in Medicine*, 36(3):257–267, 2006.

L. Oukhellou. *Paramétrisation et Classification de Signaux en Contrôle Non Destructif. Application à la Reconnaissance des Défauts de Rails par Courants de Foucault*. Thèse de doctorat, Université PARIS-SUD, Centre d'ORSAY, 1997.

D. J. Pedregal, F. P. García, and F. Schmid. RCM^2 predictive maintenance of railway systems based on unobserved components models. *Reliability Engineering and System Safety*, 83:103–110, 2004.

K. B. Petersen and M. S. Pedersen. The Matrix Cookbook, oct 2008. URL http://www2.imm.dtu.dk/pubdb/p.php?3274. Version 2008.

F. Picard, S. Robin, E. Lebarbier, and J. J. Daudin. A Segmentation/Clustering Model for the Analysis of Array CGH Data. *Biometrics*, 63(3):758–766, September 2007.

R. E. Quandt and J. B. Ramsey. Estimating mixtures of normal distributions and switching regressions. *Journal of the American Statistical Association*, 73(730-738), 1978.

L. Rabiner and B.-H. Juang. *Fundamentals of speech recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

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

J. O. Ramsay and C. J. Dalzell. Some Tools for Functional Data Analysis. *Journal of the Royal Statistical Society. Series B (Methodological)*, 53(3):539–572, 1991.

J. O. Ramsay and B. W. Silverman. *Applied Functional Data Analysis: Methods and Case Studies*. Springer Series in Statistics. Springer, 2002.

J. O. Ramsay and B. W. Silverman. *Functional Data Analysis*. Springer Series in Statistics. Springer, June 2005.

F. Rossi and B. Conan-Guez. Functional Multi-Layer Perceptron: a nonlinear tool for functional data analysis. *Neural Networks*, 18(1):45–60, January 2005a.

F. Rossi and B. Conan-Guez. Un modèle neuronal pour la régression et la discrimination sur données fonctionnelles. *Revue de Statistique Appliquée*, LIII(4):5–30, 2005b.

F. Rossi and B. Conan-Guez. Theoretical Properties of Projection Based Multilayer Perceptrons with Functional Inputs. *Neural Processing Letters*, 23(1):55–70, February 2006.

F. Rossi and N. Villa. Support Vector Machine For Functional Data Classification. *Neurocomputing*, 69(7–9):730–742, March 2006.

M.P. Ruppert, D. Wand and R.J. Carroll. *Semiparametric Regression.* Cambridge University Press, 2003.

A. Samé, C. Ambroise, and G. Govaert. An online classification EM algorithm based on the mixture model. *Statistics and Computing*, 17(3):209–218, 2007.

M-A. Sato and S. Ishii. On-line EM Algorithm for the Normalized Gaussian Network. *Neural Computation*, 12(2):407–432, February 2000.

G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

A. J. Scott and M. J. Symons. Clustering methods based on likelihood ratio criteria. *Biometrics*, 27:387–397, 1971.

A. J. Scott and M. J. Symons. Clustering Criteria and Multivariate Normal Mixtures. *Biometrics*, 37:35–43, 1981.

B. Sin and J. H. Kim. Nonstationary hidden Markov model. *Signal Processing*, 46(1): 31–46, 1995.

P. Smyth. Hidden Markov models and neural networks for fault detection in dynamic systems. In *Neural Networks for Signal Processing*, pages 582–592, 1993.

P. Smyth. Hidden Markov models for fault detection in dynamic systems. *Pattern recognition*, 27(1):149–164, 1994.

P. Smyth. Clustering Sequences with Hidden Markov Models. In *Advances in Neural Information Processing Systems 9, NIPS*, pages 648–654, 1996.

P. Smyth, U. M. Fayyad, M. C. Burl, P. Perona, and P. Baldi. Inferring Ground Truth from Subjective Labelling of Venus Images. In *NIPS*, pages 1085–1092, 1994.

C. Spearman. General intelligence, objectively determined and measured. *American Journal of psychology*, 15:201–293, 1904.

M. Stephens. *Bayesian Methods for Mixtures of Normal Distributions.* PhD thesis, University of Oxford, 1997.

M. Stephens. Bayesian Analysis of Mixture Models with an Unknown Number of Components – an alternative to reversible jump methods. *Annals of Statistics*, 28 (1):40–74, 2000.

H. Stone. Approximation of curves by line segments. *Mathematics of Computation*, 15 (73):40–47, 1961.

D. Titterington, A. Smith, and U. Makov. *Statistical Analysis of Finite Mixture Distributions*. John Wiley & Sons, 1985.

D. M. Titterington. Recursive parameter estimation using incomplete data. *Journal of the Royal Statistical Society*, 46(2):257–267, 1984.

V. N. Vapnik. *The Nature of Statistical Learning Theory (Information Science and Statistics)*. Springer, 1999.

V. N. Vapnik and V. Chervonenkis. Teoriya Raspoznavaniya Obrazov: Statisticheskie Problemy Obucheniya (Theory of Pattern Recognition: Statistical Problems of Learning), 1974.

A. J. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, 13(2):260–269, 1967.

G. Wahba. *Spline models for observational data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1990.

S. Wang and Y. Zhao. Almost sure convergence of Titterington's recursive estimator for mixture models. *Statistics & Probability Letters*, 76(18):2001–2006, December 2006.

S. R. Waterhouse. *Classification and regression using Mixtures of Experts*. PhD thesis, Department of Engineering, Cambridge University, 1997.

G. C. G. Wei and M. A. Tanner. A Monte Carlo Implementation of the EM Algorithm and the Poor Man's Data Augmentation Algorithms. *Journal of the American Statistical Association*, 85(411):699–704, 1990.

C.S. Wong and W.K. Li. On a logistic mixture autoregressive model. *Biometrika*, 88 (3):833–846, 2001.

C. F. Jeff Wu. On the Convergence Properties of the EM Algorithm. *The Annals of Statistics*, 11(1):95–103, 1983.

L. Xu and M. I. Jordan. On Convergence Properties of the EM Algorithm for Gaussian Mixtures. *Neural Computation*, 8(1):129–151, 1996.

List of Figures

List of Tables

List of Algorithms

List of publications

International journal papers

- F. Chamroukhi, A. Samé, G. Govaert and P. Aknin (2009) "Time series modeling by a regression approach based on a latent process", *Neural Networks, 22*(5-6), pp. 593-602
- F. Chamroukhi, A. Samé, G. Govaert and P. Aknin (2010) "A hidden process regression model for functional data description. Application to curve discrimination", *Neurocomputing, 73*(7-9), pp. 1210-1221

National journal paper (In French)

- F. Chamroukhi, A. Samé, G. Govaert et P. Aknin (2010) "Algorithme EM et modèle à processus latent pour la régression non liénaire" (EM algorithm and a latent process model for non-linear regression), To appear in *Revue des Nouvelles Technologies de l'Information*

International conference papers

- F. Chamroukhi, A. Samé, G. Govaert and P. Aknin (14-19 June 2009) "A regression model with a hidden logistic process for feature extraction from time series", *IEEE International Joint Conference on Neural Networks IJCNN*, Atlanta, GA, USA.
- R. Onanena, F. Chamroukhi, L. Oukhellou, D. Candusso, P. Aknin, D. Hissel (13-15 December 2009) "Supervised learning of a regression model based on a latent process. Application to the estimation of the fuel cell lifetime", *8th IEEE International Conference on Machine Learning and Applications*, ICMLA'09, Miami Beach, FL, USA.
- F. Chamroukhi, A. Samé, G. Govaert and P. Aknin (22-24 April 2009) "A regression model with a hidden logistic process for signal parametrization", *European Sympsum in Artificial Neural Networks ESANN*, Bruges, Belgium.

- F. Chamroukhi, A. Samé and P. Aknin (23-25 June 2009) “A probabilistic approach for the classification of railway switch operating states”, *The 6th International Conference on Condition Monitoring and Machinery Failure Prevention Technologies*, Dublin, UK.
- F. Chamroukhi, A. Samé, P. Aknin and M. Antoni (18-20 June 2008) “Switch mechanism diagnosis using a pattern recognition approach”, *The 4th IET International Conference on Railway Condition Monitoring RCM 2008*, Derby, UK.

National conference papers (in French)

- A. Samé , F. Chamroukhi and G. Govaert (25-29 May 2009) “Algorithme EM et modèle à processus latent pour la régression non liénaire” (EM algorithm and a latent process model for non-linear regression), *41st meeting of the French Statistical Society*, Bordeaux, France.
- F. Chamroukhi, A. Samé, P. Aknin and G. Govaert (Dec. 2008) “A regression model with latent variable for modeling the railway switch mechanism signals”, *INRETS annual Workshop*, Collection Number 119, ISSN 0769-0266, pp. 57-64

Reports

- F. Chamroukhi “Pattern recognition for the diagnosis of the French railway switches” (June 2007), Report for INRETS and the French railway company (SNCF) and Master report (In French),

Titre : Régression à processus latent pour la modélisation, la classification et le suivi de courbes

Résumé : Cette thèse s'est focalisée sur l'analyse de courbes à changements de régime. Nous proposons de nouvelles approches probabilistes génératives pour modéliser, classer et suivre temporellement de telles courbes. Le premier volet de la thèse concerne la modélisation et la classification (supervisée ou non) d'un ensemble de courbes indépendantes. Les approches proposées dans ce cadre, qui peuvent être appliquées aussi bien à une courbe qu'à un ensemble de courbes, reposent sur un modèle de régression spécifique incorporant un processus caché s'adaptant aussi bien à des changements de régimes brusques qu'à des changements lents. Le second volet de la thèse concerne la modélisation dynamique d'une séquence de courbes à changements de régime. Nous proposons pour cela des modèles autorégressifs intégrant eux même un processus caché et dont l'apprentissage est réalisé à la fois en mode "hors ligne", quand les courbes sont stockées à l'avance, et en mode "en ligne", quand les courbes arrivent au fur et à mesure au cours du temps. Le volet applicatif de la thèse concerne le diagnostic et le suivi d'état de fonctionnement du mécanisme d'aiguillage des rails qui est un organe impactant considérablement la disponibilité du réseau ferroviaire. Sa surveillance est essentielle pour mieux planifier les actions de maintenance. Les données disponibles pour réaliser cette tâche sont les courbes de puissance électrique acquises lors des manœuvres d'aiguillage, qui ont notamment la particularité de présenter des changements de régime. Les résultats obtenus sur des courbes simulées et des courbes acquises lors de manœuvres d'aiguillage illustrent l'utilité pratique des approches introduites dans cette thèse.

Mots-clés : Modélisation de courbes, régression, classification, modèle de mélange, Modèle de Markov caché, modélisation dynamique, apprentissage en ligne, algorithmes EM, diagnostic.