Contributions to the Bayesian Analysis of Mixture Models

a thesis submitted to

The University of Kent at Canterbury

in the subject of Statistics

for the degree

of Doctor of Philosophy

by

© Carlos Erwin Rodríguez Hernández-Vela

January 13, 2013

Abstract

Mixture models can be used to approximate irregular densities or to model heterogeneity. When a density estimate is needed, then we can approximate any distribution on the real line using an infinite number of normals (Ferguson (1983)). On the other hand, when a mixture model is used to model heterogeneity, there is a proper interpretation for each element of the model. If the distributional assumptions about the components are met and the number of underlying clusters within the data is known, then in a Bayesian setting, to perform classification analysis and in general component specific inference, methods to undo the label switching and recover the interpretation of the components need to be applied. If latent allocations for the design of the Markov chain Monte Carlo (MCMC) strategy are included, and the sampler has converged, then labels assigned to each component may change from iteration to iteration. However, observations being allocated together must remain similar, and we use this fundamental fact to derive an easy and efficient solution to the label switching problem. We compare our strategy with other relabeling algorithms on univariate and multivariate data examples and demonstrate improvements over alternative strategies.

When there is no further information about the shape of components and the number of clusters within the data, then a common theme is the use of the normal distribution as the "benchmark" components distribution. However, if a cluster is skewed or heavy tailed, then the normal distribution will be inefficient and many may be needed to model a single cluster. In this thesis, we present an attempt to solve this problem. We define a cluster to be a group of data which can be modeled by a unimodal density function. Hence, our intention is to use a family of univariate distribution functions, to replace the normal, for which the only constraint is unimodality. With this aim, we devise a new family of nonparametric unimodal distributions, which has large support over the space of univariate unimodal distributions. The difficult aspect of the Bayesian model is to construct a suitable MCMC algorithm to sample from the correct posterior distribution. The key will be the introduction of strategic latent variables and the use of the product space (Godsill (2001)) view of reversible jump (Green (1995)) methodology. We illustrate and compare our methodology against the classic mixture of normals using simulated and real data sets. To solve the label switching problem we use the new relabeling algorithm.

Acknowledgements

I would like to thank my supervisor, Professor Stephen G. Walker, who contributed his time and ideas to this thesis. I must also give thanks to him for his skillful supervision, and giving me continued support and encouragement when completing this work. Thanks also to Eduardo Gutierrez-Peña for bringing me in touch with Professor Walker and to all the researchers of the Department of Probability and Statistics at IIMAS-UNAM. Special thanks to Silvia Ruiz Velasco for believing in my abilities and Raúl Rueda for his support and great sense of humour. The financial support of CONACYT, the Mexican National Council for Science and Technology is gratefully acknowledge.

I also would like to thank my family and friends for all that they have done for me over these years.

Finally, I must pay a special thanks to my wife, Claudia López Soto, for her constant support, for being there for me in the good times and also when times were really hard. This would not have been possible without you. I love you.

Notation

a. s. - denotes almost surely or with probability 1

i. i. d. - stands for independent and identically distributed

$\mathbb{1}_A(\cdot)$ - denotes the indicator function: $\mathbb{1}_A(x) = \begin{cases} 1, & \text{if } x \in A, \\ 0, & \text{if } x \notin A. \end{cases}$

$\delta_x(\cdot)$ - denotes the measure giving unit mass to the point x, $\delta_x(A) = \mathbb{1}_A(x)$

$\mathbb{R}$ - denotes the set of real numbers, and in general $\mathbb{R}^k$ will denote a general Euclidean space

$\mathbb{N}$ - denotes the set of positive integer numbers

$\mathscr{P}$ - denotes the Dirichlet Process measure with base distribution G_0 and concentration parameter c.

$\text{supp}(f)$ - denotes the support of the function f, i.e. if $f : X \to \mathbb{R}$, then $\text{supp}(f) = \{x \in X | f(x) \neq 0\}$

$\mathbb{E}(\cdot)$ - denotes the expectation operator of a random variable

$\text{var}(\cdot)$ - denotes the variance operator of a random variable

$\mathbf{Pr}(A)$ - denotes the probabilty of the event A

$\mathscr{B}(\mathbb{R}^k)$ - denotes the Borel σ algebra on the Euclidean space $\mathbb{R}^k$

$\Gamma(\cdot)$ - denotes the gamma function

$B(\cdot, \cdot)$ - denotes the beta function, i.e. $B(x, y) = \dfrac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)}$

$\sim$ - used to stand that X is distributed by G, i.e. $X \sim G$

$y_{(1)}$ - denotes the minimum over a set of n observations, and we will denote the maximum as $y_{(n)}$

R - denotes the range of the data, i.e. if we have a sample of n observations, $R = y_{(n)} - y_{(1)}$

Probability Distributions

$\text{Be}(\alpha, \beta)$ - denotes the beta distribution

$\text{Dir}(\alpha_1, ..., \alpha_k)$ - denotes the Dirichlet distribution

$\text{Ga}(\alpha, \beta)$ - denotes the gamma distribution with mean α/β

$\text{Inv-Ga}(\alpha, \beta)$ - denotes the inverse gamma distribution

$\text{N}(\mu, \sigma^2)$ - denotes the normal distribution; the p-dimension version is denoted by $\text{N}_p(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is a $p \times p$ positive definite matrix

$\text{Po}(\lambda)$ - denotes the Poisson distribution

$\text{U}(a, b)$ - denotes the uniform distribution on $[a, b]$

$\text{Wi}_p(m, A)$ - denotes the Wishart distribution with positive definite scale matrix $A_{p\times p}$ and $m > p - 1$

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

Mixture models are widely used in many scientific fields. They provide a flexible tool to study data sets arising from irregular densities or to model heterogeneity. The first known attempt to model heterogeneity, under a mixture modeling setting, can be traced back to Pearson (1894). While analyzing measurements of the ratio of forehead to body length of crabs, Pearson had evidence to believe that there were two species present within the sample. He then used the method of moments to estimate the parameters of a mixture of two univariate normals.

Nowadays, to fit a mixture model, the preferred methods rely on intensive computing techniques. On the one hand, to find maximum likelihood estimators for the parameters of a mixture model, the Expectation Maximization (EM) algorithm, Dempster, Laird, and Rubin (1977) is used. On the other hand, in a Bayesian setting, a sample from the posterior distribution of the model is generated via Markov chain Monte Carlo methods (MCMC), Gelfand and Smith (1990) and Smith and Roberts (1993), and then posterior distribution summaries are calculated from the MCMC output. A comprehensive review of Bayesian and Frequentist methods for mixture models can be found in Frühwirth-Schnatter (2006).

From the Bayesian point of view there are two types of mixture models: One assumes

there is a finite integer k, which is the number of mixtures required to model the data, and k is assumed known or unknown and can take any positive integer value. This is best known as the finite mixture model with k components and is written as

$$p(y_i|\mathbf{w}, \boldsymbol{\theta}, k) = \sum_{j=1}^{k} w_j f(y_i|\theta_j), \text{ independently for } i = 1, \ldots, n, \tag{1.1}$$

and let $\boldsymbol{\theta} = \{\theta_j\}_{j=1}^k$ and $\mathbf{w} = \{w_j\}_{j=1}^k$. For (1.1) to be a density, the weights, $\mathbf{w}$, must be non-negative and sum to one. A useful idea when working with model (1.1) is to include latent allocation variables: $\mathbf{z} = \{z_i\}_{i=1}^n$, such that given z_i, the component from which y_i has been drawn is known, i.e.

$$p(y_i|\boldsymbol{\theta}, z_i) = f(y_i|\theta_{z_i}). \tag{1.2}$$

There is a proper interpretation for (1.1): k represents the number of clusters within the data, w_j the weight of cluster j in the population and $f(y|\theta_j)$ a parametric distribution that models the behavior of cluster j. The latent allocations (1.2) automatically induce a clustering structure over the observations.

The alternative model consists in mixing a kernel $k(y|\theta)$ with respect to a random distribution function G to define a random density:

$$f_G(y) = \int_\Theta k(y|\theta) G(\mathbf{d}\theta) = \sum_{j=1}^{\infty} w_j k(y|\theta_j). \tag{1.3}$$

Since the prior over G is usually taken as a discrete measure, the infinite mixture (1.3) is derived indirectly. Under this setting, the main aim is density estimation (Lo (1984)), and there is no explicit parameter modeling the number of groups or clusters, and in general there is no clear interpretation for the parameters of (1.3). There is no infinite number of clusters when $w_j > 0$ for all j. However, it is possible to recover a meaning for a small subset of the parameters.

Algorithms to perform Bayesian analysis of the finite mixture model, (1.1), first assumed k to be known and were later extended to cover the k unknown case. For a fixed k, one of the first works using MCMC is given in Diebolt and Robert (1994). To

make inference for an unknown number of components, Richardson and Green (1997) used reversible jump MCMC ideas (Green (1995)), and Stephens (2000a) used a birth-death process. More recent work done by Nobile and Fearnside (2007) is based on a variation of the model and the sampling technique requires that the parameters of the model can be integrated out analytically. An interesting discussion and review on trans-dimensional MCMC methods can be found in Green (2003) and Sisson (2005).

For the latter mixture model, (1.3), when the prior over G is taken as the Dirichlet process (Ferguson (1973)), it is possible to derive a finite model by integrating out the random distribution; see, for example, Escobar (1988), Escobar (1994) and Escobar and West (1995). More recent approaches utilize the constructive definition of the Dirichlet process, Sethuraman (1994), and work directly with the infinite number of mixtures. Appropriate algorithms then need to be constructed so that the correct posterior is sampled by knowing how many of the infinite variables need to be drawn. See Papaspiliopoulos and Roberts (2008) for a retrospective sampler, and Walker (2007), and Kalli, Griffin, and Walker (2011) for slice samplers.

1.1 Label switching

If the distributional assumptions about the components, in (1.1), are met and the number of underlying clusters within the data is known, then to perform classification analysis or in general inference for any characteristic of the components, the objective is to find the posterior distribution which is approximated via MCMC methods. A problem emerges when we note that the likelihood of (1.1) is invariant under permutation of the indices: and there are $k!$ permutations. Thus, under symmetric priors, the posterior will inherit the likelihood's invariance. As a result, in any MCMC algorithm, labels of the components can permute multiple times between iterations of the sampler. This makes ergodic averages to estimate characteristics of the components useless. This is known as the label switching problem.

Paradoxically, as noted by Celeux, Hurn, and Robert (2000), Frühwirth-Schnatter (2001), Jasra, Holmes, and Stephens (2005) and Papastamoulis and Iliopoulos (2010), among others, label switching is a prerequisite for MCMC convergence. If there is no label switching, then it means that the sampler is not exploring all the modes of the posterior distribution of (1.1). It should visit all the $k!$ symmetric modes to cover the whole posterior span. Notably, when the modes of the posterior are well separated, the standard Gibbs sampler fails the label switching test (Jasra, Holmes, and Stephens (2005) p. 55) as it becomes trapped in one of the symmetric modes. While this may be meaningful for inference, it becomes impossible to justify convergence.

To address this problem, three alternative ideas have been suggested. The first one keeps the standard Gibbs sampler and incorporates a Metropolis-Hastings move that proposes a random permutation of the labels. See Frühwirth-Schnatter (2001) and Papastamoulis and Iliopoulos (2010). The second approach is to use more sophisticated and complex MCMC methods to improve mixing of the sampler, and specifically avoiding the use of (1.2). See Celeux, Hurn, and Robert (2000) and Jasra, Holmes, and Stephens (2005). The third idea is to use a trans-dimensional sampler, see Jasra, Holmes, and Stephens (2005).

Initial attempts to "undo the label switching" were focused on imposing artificial constraints on the parameter space via the prior distribution aiming to break the symmetry of the posterior distribution and force a unique labeling. See Diebolt and Robert (1994) or Richardson and Green (1997). However, Stephens (1997) showed that these constraints do not always solve the problem. Another idea, based on identifiability constraints over the parameter space, was proposed by Frühwirth-Schnatter (2001). The main criticism for this method is the difficulty to select the adequate constraints among all the possible choices.

More elaborate solutions aim to undo the label switching deterministically. This means finding permutations of the parameters that minimize a loss function. See Celeux

(1998), Stephens (1997), Nobile and Fearnside (2007) and Grün and Leisch (2009). Recent methods combine similar ideas with the use of the maximum a posteriori (MAP) estimator. See Martin, Mengersen, and Robert (2005) and Papastamoulis and Iliopoulos (2010). Other approaches focus on devising a loss function invariant to permutations of the parameters, see for example Celeux, Hurn, and Robert (2000), thus, solving the label switching problem immediately. The drawback of this method is that it is computationally expensive and to define such a loss function is not always possible, see Celeux, Hurn, and Robert (2000) and Jasra, Holmes, and Stephens (2005).

In this thesis we propose a solution to the label switching problem, Rodríguez and Walker (2013), that lies in the meaning of the relationship between the allocation variables (1.2) and the observations. The key is to use this relationship to incorporate the data directly within the loss function used to undo the label switching. From iteration to iteration of an MCMC algorithm the labels of the clusters may change. But if the sampler has converged, the clusters must remain roughly the same. We use this fundamental fact to derive an easy and efficient solution to the label switching problem. To asses the effect of the chosen MCMC strategy on the resulting inference, we compare results obtained via the standard Gibbs sampler against those obtained via a trans-dimensional MCMC algorithm.

1.2 Modelling of clusters

A common theme in (1.1) and (1.3), is the use of the normal distribution as the "benchmark" components or kernel distribution. This has been mainly for two reasons; it is a well known distribution and when used with conjugate priors the resulting MCMC simplifies. Under the finite mixture set-up, attempts to work with other parametric components distributions are few: see, for example, Stephens (2000a), who used the Student's-t distributions, and Wiper, Rios-Insua, and Ruggeri (2001), who used the gamma distribution. Within the Bayesian nonparametric literature, unimodal distribu-

tions have been explored; see for example Lo (1984) and Brunner and Lo (1989), but there has been no attempt to incorporate such a nonparametric unimodal distribution as the components distribution in a mixture model.

If a density estimate is needed then the use of the normal distribution is perfectly justified. We can approximate any distribution on the real line using an infinite mixture of normals (Ferguson (1983)). However, for the modeling of clusters, it does have some serious issues: if a cluster is skewed or heavy tailed then the normal distribution will be inefficient and many may be needed to model a single cluster. To motivate our proposal we can cite two important works in Bayesian mixture modeling:

Escobar and West (1995), when analyzed the galaxy data using mixtures of Dirichlet processes and obtained unrealistic high posterior values for the number of components.

> "The underlying assumption is that each galactic cluster is a normal component. If the distribution of a galactic cluster is skewed or has a very light or heavy tail, then we may use two or more normal components to fit one galactic cluster component."

Richardson and Green (1997), while analyzing three data sets using a finite mixture of normals observed the same problem as Escobar and West

> "In each case, the high overall number of components can be related in part to the skewness of the data, two or three normals being sometimes needed to fit one skewed component."

Hence, to model a single cluster, two or possibly more normals are needed, so the number of components does not coincide with the number of clusters; meaning that the number of components needed to model the data via a mixture of normal distributions, has no real interpretation.

In this thesis our aim is to ensure the number of clusters within the data coincides with the estimate of the number of components, Rodríguez and Walker (2012). Our

plan is to use a components distribution for which the only constraint is unimodality. We employ this in a finite mixture model context, (1.1), where k is modeled explicitly. With this objective, we introduce a new family of nonparametric unimodal distributions, which has large support over the space of unimodal distributions. Hence, the model is a finite mixture of k unimodal densities, each of which is modeled nonparametrically. In short, therefore, we are defining a cluster as a set of observations which can be modeled by a unimodal density. In the absence of further information beyond the observations, this is the most reasonable working assumption for a cluster. The idea being that a multimodal density would reasonably be assumed to contain more than one cluster.

The thesis is structured as follows:

Chapter 2: provides an outline of MCMC methods for finite mixtures. Emphasis is given to the case for an unknown number of components, where the product space model of Godsill (2001) is used to derive the reversible jump acceptance probability for the finite mixture model (1.1). Under this setting, a complete summary of Richardson and Green (1997) ideas is given.

Chapter 3: describes the properties of the Dirichlet process and provides an overview of some MCMC methods to sample from the mixture of Dirichlet process model. The aim of this chapter is to motivate the ideas rather than provide a detailed account of theoretical results.

Chapter 4: delineates a new and efficient solution to the label switching problem: a deterministic relabeling algorithm. First, the key ideas for a deterministic relabeling algorithm are described and from it a solution to the label switching problem is proposed. The new strategy is compared with other relabeling algorithms on univariate and multivariate data examples.

Chapter 5: defines a new family of nonparametric unimodal distributions and a finite mixture of k of these unimodal distributions. Then, it describes an MCMC

algorithm to sample from the posterior distribution of this finite mixture model: known and unkown k cases are treated. Real and simulated data sets are analyzed with the new model and compared with the analysis performed via the mixture of normal distributions.

Appendix A: provides a background on the basic definitions and ideas of Markov chains and the MCMC methods as used in this thesis.

Chapter 2

MCMC Methods for Finite Mixtures

In this Chapter, MCMC methods to fit a finite mixture model are reviewed. Special attention is given to the case when the number of components is unknown, hence we formulate the model under this assumption. However, it is easier to deal with the problem of a known number of components first, and then extend the sampler to include the case for a moving number of components. This is standard procedure. For the finite mixture of normals, Diebolt and Robert (1994) first devised the case for a fixed number of components, now a very well known Gibbs sampler, and then later other authors used this as a corner-stone to derive algorithms for normal distributions with an unknown number of components; see for example Richardson and Green (1997) and Stephens (2000b).

2.1 The model

The finite mixture model with k components, and k is assumed unknown, is written as

$$p(y_i|\mathbf{w}, \boldsymbol{\theta}, k) = \sum_{j=1}^{k} w_j f(y_i|\theta_j), \text{ independently for } i = 1, \ldots, n, \tag{2.1}$$

and let $\boldsymbol{\theta} = \{\theta_j\}_{j=1}^k$ and $\mathbf{w} = \{w_j\}_{j=1}^k$. For (2.1) to be a density, the weights, $\mathbf{w}$, must be non-negative and sum to one. A clever idea that simplifies many calculations when working with (2.1) is to include the latent allocation variables: $\mathbf{z} = \{z_i\}_{i=1}^n$, such that given z_i, the component from which y_i has been drawn is known, i.e.

$$p(y_i|\boldsymbol{\theta}, z_i) = f(y_i|\theta_{z_i}). \tag{2.2}$$

Note, a *priori*, each z_i is drawn independently with distribution $p(z_i = j|\mathbf{w}, k) = w_j$, for $j = 1, \ldots, k$. Integrating out z_i, we return to (2.1), since

$$\sum_{j=1}^{k} p(z_i = j|\mathbf{w}, k)p(y_i|\boldsymbol{\theta}, z_i = j) = \sum_{j=1}^{k} w_j f(y_i|\theta_j).$$

Now let $n_j = \#\{i : z_i = j\}$, so

$$p(\mathbf{z}|\mathbf{w}, k) \propto \prod_{i=1}^{n} w_{z_i} = \prod_{j=1}^{k} w_j^{n_j}. \tag{2.3}$$

This is the multinomial distribution, and it is well defined if $n_j = 0$ for some j (see Section 3.1.1 in Chapter 3).

Also, with the introduction of the latent variables (2.2), the likelihood of (2.1), can be written as

$$p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{z}) = \prod_{i=1}^{n} f(y_i|\theta_{z_i}), \tag{2.4}$$

where $\mathbf{y} = \{y_i\}_{i=1}^n$.

2.1.1 Hierarchical model and priors

In a Bayesian setting the unknowns k, $\mathbf{w}$ and $\boldsymbol{\theta}$ are treated as random variables, so to learn about them we specify prior beliefs via prior distributions: $p(k|\lambda)$, $p(\mathbf{w}|\delta)$ and $p(\boldsymbol{\theta}|\gamma)$ where λ, δ and γ are constants. Or to gain flexibility, additional hierarchical levels can be added to λ, δ and γ. In our case, we will only include an extra hierarchical level to γ: $p(\gamma|\zeta)$. For the latent variables $\mathbf{z}$, the prior, (2.3), is specified indirectly. With

these considerations it is possible to write a hierarchical representation of the model:

$$
\begin{aligned}
&[k|\lambda] \sim p(\cdot|\lambda) \quad \text{and} \quad [\gamma|\zeta] \sim p(\cdot|\zeta), \\
&[\mathbf{w}|\delta, k] \sim p(\cdot|\delta, k), \\
&[\theta_j|\gamma, k] \sim p(\cdot|\gamma), \ j = 1, 2, \ldots, k, \\
&[z_i|\mathbf{w}, k] \sim \sum_{j=1}^{k} w_j \delta_j(\cdot), \quad i = 1, \ldots, n, \\
&[y_i|\boldsymbol{\theta}, \mathbf{z}] \sim f(\cdot|\theta_{z_i}), \quad i = 1, \ldots, n.
\end{aligned} \tag{2.5}
$$

Then, using Bayes' Theorem, we update knowledge via the posterior distribution:

$$
\begin{aligned}
p(k, \gamma, \boldsymbol{\theta}, \mathbf{w}, \mathbf{z}|\mathbf{y}) &\propto p(k, \gamma, \boldsymbol{\theta}, \mathbf{w}, \mathbf{z}, \mathbf{y}) \\
&= p(k|\lambda)p(\gamma|\zeta)p(\boldsymbol{\theta}|\gamma, k)p(\mathbf{w}|\delta, k)p(\mathbf{z}|\mathbf{w}, k)p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{z}).
\end{aligned} \tag{2.6}
$$

Note that some conditional independence has been assumed: from (2.2) we already had $p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{z}, \mathbf{w}, k) = p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{z})$ and now further $p(\boldsymbol{\theta}|\mathbf{z}, \mathbf{w}, k) = p(\boldsymbol{\theta}|k)$ is included.

The prior for the number of groups, $p(k|\lambda)$, is taken as discrete uniform, if we want to be non-informative, or truncated Poisson if there exists some prior knowledge about the number of groups. The prior for the weights is often taken as a symmetric Dirichlet distribution: $p(\mathbf{w}|\delta, k) = \text{Dir}(\mathbf{w}|\delta, \ldots, \delta)$. We will use this specification throughout for the weights of the finite mixture model. The $\boldsymbol{\theta} = \{\theta_j\}_{j=1}^{k}$, are assumed to be drawn independently, i.e.

$$
p(\boldsymbol{\theta}|\gamma, k) = \prod_{j=1}^{k} p(\theta_j|\gamma).
$$

Finally, for $p(\theta_j|\gamma)$ and $p(\gamma|\zeta)$ conjugate priors are usually assigned.

2.2 Known number of components

When k is known, it is straightforward to devise a Gibbs sampler (see Algorithm A.4.1 in Appendix A) to sample from the posterior distribution (2.6). Note from (2.6) that

the full joint conditionals are proportional to

$$p(\mathbf{z}|\mathbf{w},k)p(\mathbf{y}|\boldsymbol{\theta},\mathbf{z}) = \prod_{i=1}^{n} w_{z_i} f(y_i|\theta_{z_i}) = \prod_{j=1}^{k} \left\{ w_j^{n_j} \prod_{\{i:z_i=j\}} f(y_i|\theta_j) \right\}. \tag{2.7}$$

A description of this strategy is displayed in Algorithm 2.1 (note that $n_j^t = \#\{i : z_i^t = j\}$).

Algorithm 2.1 Gibbs sampler for finite mixtures

Require: Given $(\mathbf{w}^t, \boldsymbol{\theta}^t, \mathbf{z}^t, \gamma^t)$, simulate $(\mathbf{w}^{t+1}, \boldsymbol{\theta}^{t+1}, \mathbf{z}^{t+1}, \gamma^{t+1})$ via

1: **for** $j = 1$ to k **do**

2: $\quad \theta_j^{t+1} \sim p(\theta_j|\mathbf{z}^t, \gamma^t, \mathbf{y}) \propto p(\theta_j|\gamma^t) \prod_{\{i:z_i^t=j\}} f(y_i|\theta_j).$

3: **end for**

4: $\mathbf{w}^{t+1} \sim p(\mathbf{w}|\delta, \mathbf{z}^t, k) \propto p(\mathbf{w}|\delta, k) \prod_{j=1}^{k} w_j^{n_j^t} \propto \mathrm{Dir}(\mathbf{w}|n_1^t + \delta, \ldots, n_k^t + \delta).$

5: **for** $i = 1$ to n **do**

6: $\quad z_i^{t+1} \sim p(z_i = j|\mathbf{w}^{t+1}, \boldsymbol{\theta}^{t+1}, k, \mathbf{y}) = \dfrac{w_j^{t+1} f(y_i|\theta_j^{t+1})}{\sum_{j=1}^{k} w_j^{t+1} f(y_i|\theta_j^{t+1})} \quad (j = 1, \ldots, k).$

7: **end for**

8: $\gamma^{t+1} \sim p(\gamma|\zeta, \boldsymbol{\theta}^{t+1}, k) \propto p(\gamma|\zeta) \prod_{j=1}^{k} p(\theta_j^{t+1}|\gamma).$

Algorithm 2.1 was first used by Diebolt and Robert (1994), however additional hierarchical levels have been included (motivated by Richardson and Green (1997)).

2.2.1 Poisson mixtures

We can easily find the full conditionals for the case when the Poisson distribution is taken as the components distribution: $f(y_i|\theta_j) = \mathrm{Po}(y_i|\theta_j)$. Here independent gamma priors for the means can be assumed: $\gamma = (\alpha, \beta) \Rightarrow p(\theta_j|\gamma) = \mathrm{Ga}(\theta_j|\alpha, \beta)$, then to sample $\theta_j^{t+1} \sim p(\theta_j|\mathbf{z}^t, \gamma^t, \mathbf{y})$, in Algorithm 2.1, we draw

$$\theta_j^{t+1} = p(\theta_j|\alpha, \beta, \mathbf{z}^t, \mathbf{y}) \sim \mathrm{Ga}\left(\alpha + \sum_{\{i:z_i^t=j\}} y_i, \beta + n_j^t \right).$$

We will not assume further hierarchical structures for $\gamma = (\alpha, \beta)$.

2.2.2 Univariate normal mixtures

If the normal distribution is taken as the components distribution: $\theta_j = (\mu_j, \tau_j^{-1})$, hence

$$f(y_i|\theta_j) = \mathrm{N}(y_i|\mu_j, \tau_j^{-1}), \quad \text{with} \quad \tau_j^{-1} = \sigma_j^2.$$

For this model we will differentiate between the known and unknown k case:

- Known k. The usual choices of priors are independent normal and gamma distributions for the means and precisions, respectively. Hence

  $$p(\theta_j|\gamma) = \mathrm{N}(\mu_j|\mu_0, \kappa^{-1})\,\mathrm{Ga}(\tau_j|\alpha, \beta),$$

  where $\gamma = (\mu_0, \kappa^{-1}, \alpha, \beta)$ (the $\mathrm{Ga}(\cdot|\alpha, \beta)$ is always parametrized such that the mean is given by α/β). In this case an additional hierarchical level for β is assumed, thus $p(\beta|g, h) = \mathrm{Ga}(\beta|g, h)$. With this consideration, to sample $\theta_j^{t+1} \sim p(\theta_j|\mathbf{z}^t, \gamma^t, \mathbf{y})$ and $\gamma^{t+1} \sim p(\gamma|\zeta, \boldsymbol{\theta}^{t+1}, k, \mathbf{y})$ in Algorithm 2.1, we generate

  $$\begin{aligned}
  \mu_j^{t+1} &\sim p(\mu_j|\mu_0, \kappa, \tau_j^t, \mathbf{z}^t, \mathbf{y}) = \mathrm{N}\left(\frac{\tau_j^t \sum_{\{i:z_i^t=j\}} y_i + \kappa\mu_0}{\tau_j^t n_j^t + \kappa}, \frac{1}{\tau_j^t n_j^t + \kappa}\right), \\
  \tau_j^{t+1} &\sim p(\tau_j|\alpha, \mu_j^{t+1}, \beta^t, \mathbf{z}^t, \mathbf{y}) = \mathrm{Ga}\left(\alpha + \frac{n_j^t}{2}, \beta^t + \frac{1}{2}\sum_{\{i:z_i^t=j\}}\left(y_i - \mu_j^{t+1}\right)^2\right), \\
  \beta^{t+1} &\sim p(\beta|g, h, \alpha, \tau^{t+1}, k, \mathbf{y}) = \mathrm{Ga}\left(g + k\alpha, h + \sum_{j=1}^k \tau_j^{t+1}\right).
  \end{aligned}$$

- Unknown k. To derive the trans-dimensional case as in Richardson and Green (1997) we change the prior for the means, i.e.

  $$p(\mu_1, \ldots, \mu_k|\mu_0, \kappa^{-1}, k) = k! \prod_{j=1}^k \mathrm{N}(\mu_j|\mu_0, \kappa^{-1})\,\mathbb{1}\{\mu_1 < \ldots < \mu_k\} \tag{2.8}$$

  the order statistics of k normal distributions. With this change, the full conditional for μ_j^{t+1} is a truncated normal distribution, in form

  $$\mu_j^{t+1} \sim p(\mu_j|\mu_0, \kappa, \tau_j^t, \mathbf{z}^t, \mathbf{y}) = \mathrm{N}\left(\frac{\tau_j^t \sum_{\{i:z_i^t=j\}} y_i + \kappa\mu_0}{\tau_j^t n_j^t + \kappa}, \frac{1}{\tau_j^t n_j^t + \kappa}\right)\mathbb{1}\left\{\mu_{j-1}^{t+1} < \mu_j < \mu_{j+1}^t\right\},$$

where $\mu_0^{t+1} = -\infty$ and $\mu_{k+1}^t = \infty$.

We stress that (2.8) is not to provide an identifiability constraint and break the symmetry of the likelihood of (2.1). It has been shown, Stephens (2000b), that these constraints do not solve the label switching problem. The purpose of this prior is to impose the order needed on the location parameters to construct the invertible transformation as in Richardson and Green (1997). Their transformation takes μ_j with $\mu_{j-1} < \mu_j < \mu_{j+1}$ and splits it into μ_{j_1} and μ_{j_2} such that $\mu_{j-1} < \mu_{j_1} < \mu_{j_2} < \mu_{j+1}$. For the inverse transformation we need to select μ_{j_1} and μ_{j_2} and combine them into μ_j, preserving the same order as in the split. See Section 2.3.2. All the other specifications remain as in the fixed k case.

Remark 2.1. The hyper-prior for β gives more flexibility to model the size of the components variance and becomes very important when making inference for the number of groups: the estimated number of components is related to the prior information on the variances σ_j^2. When performing a sensitivity analysis Richardson and Green (1997) noted that the results of the model with the additional hierarchical level were more robust to prior assumptions than to those of the model without the increased flexibility. However, for the known k case is often also included. Many authors have used this model as a benchmark for their own proposals, see for example Stephens (2000a) or Papastamoulis and Iliopoulos (2010).

2.2.3 Multivariate normal mixtures

Later on in the thesis we will work using examples with mixtures of bivariate normals. For completeness, we include the multivariate extension of the Gibbs sampler for the univariate case. Here $\theta_j = (\mu_j, \Sigma_j)$ thus $f(y_i|\theta_j) = \mathrm{N}_p(y_i|\mu_j, \Sigma_j)$, and the usual choices of priors are independent multivariate normal and Wishart (replaces the gamma as the multivariate version of a conjugate prior) distributions for the means and variance-covariance

matrices, respectively. Hence,

$$p(\theta_j|\gamma) = \mathrm{N}_p\left(\mu_j|\mu_0, \kappa^{-1}\right) \mathrm{Wi}_p\left(\Sigma_j^{-1}|2\alpha, (2\beta)^{-1}\right).$$

The equivalent hyper-prior for β, to the one of the univariate mixture of normals, becomes $p(\beta|g,h) = \mathrm{Wi}_p\left(\beta|2g, (2h)^{-1}\right)$. Thus to sample from the full conditionals, in Algorithm 2.1, we draw

$$\begin{aligned}
\mu_j^{t+1} &\sim p(\mu_j|\mu_0, \kappa, \Sigma_j^t, \mathbf{z}^t, \mathbf{y}) = \mathrm{N}_p\left(\mathrm{B}_j^t\left(\Sigma_j^{t\,-1}\sum_{\{i:z_i^t=j\}} y_i + \kappa\mu_0\right), \mathrm{B}_j^t\right), \\
\Sigma_j^{t+1\,-1} &\sim p(\Sigma_j^{-1}|\alpha, \mu_j^{t+1}, \beta^t, \mathbf{z}^t, \mathbf{y}) = \mathrm{Wi}_p\left(2\alpha + n_j^t, C_j^t\right), \\
\beta^{t+1} &\sim p(\beta|g,h,\alpha,\Sigma_j^{t+1}, k, \mathbf{y}) = \mathrm{Wi}_p\left(2g + 2k\alpha, \left\{2h + \sum_{j=1}^k \Sigma_j^{t+1\,-1}\right\}^{-1}\right),
\end{aligned}$$

where $\mathrm{B}_j^t = \left(n_j^t \Sigma_j^{t\,-1} + \kappa\right)^{-1}$ and $C_j^t = \left\{2\beta^t + \sum_{\{i:z_i^t=j\}} (y_i - \mu_j^{t+1})(y_i - \mu_j^{t+1})^T\right\}^{-1}$.

This multivariate mixture model was used by Stephens (1997) when demonstrating his relabeling algorithm, later on in the thesis we will use this model with the same purpose.

2.2.4 Convergence

If the full conditionals in Algorithm 2.1 are positive, then the Markov chain generated via the standard Gibbs sampler is ergodic; see Appendix A. This is the case for the examples that we have presented. Thus, we can approximate posterior means of relevant quantities with ergodic averages of sample paths of the chain. For example, we can approximate

$$\begin{aligned}
p(y^*|\mathbf{y}, k) &= \int_\Theta \int_W p(y^*, \mathbf{w}, \boldsymbol{\theta}|\mathbf{y}, k) d\mathbf{w} d\boldsymbol{\theta}, \qquad (2.9) \\
&= \int_\Theta \int_W p(y^*|\mathbf{w}, \boldsymbol{\theta}, k) p(\mathbf{w}, \boldsymbol{\theta}|\mathbf{y}, k) d\mathbf{w} d\boldsymbol{\theta}, \\
&= \int_\Theta \int_W \left\{\sum_{j=1}^k w_j f(y^*|\theta_j)\right\} p(\mathbf{w}, \boldsymbol{\theta}|\mathbf{y}, k) d\mathbf{w} d\boldsymbol{\theta}, \\
&\approx \frac{1}{N}\sum_{t=1}^N \left\{\sum_{j=1}^k w_j^t f(y^*|\theta_j^t)\right\} \text{(for } N \text{ large enough)},
\end{aligned}$$

where $(\mathbf{w}^t, \boldsymbol{\theta}^t)$ for $t = 1, \ldots, N$ are generated via Algorithm 2.1 (N is the number of iterations after a burn in period).

Note that in (2.9) y^* and $\mathbf{y}$ are conditionally independent. Hence, under this condition, we can generate a grid of points, $y_1^*, \ldots, y_m^*$, over the range of $\mathbf{y}$, to produce a density estimate via $(y_l^*, p(y_l^*|\mathbf{y}, k))$ for $l = 1, \ldots, m$.

Remark 2.2. The density estimate (2.9) is invariant to permutations of the mixture components', hence it is not affected by the label switching phenomenon. However, to approximate posterior means relevant to specific components of the mixture, e.g. classification probabilities. The label switching problem must be addressed first, see Chapter 4 for a complete description of the label switching problem.

2.2.5 Additional comments

The standard Gibbs sampler for finite mixtures was first described by Diebolt and Robert (1994) and since then, there have been constant concerns about the "practical convergence properties" of the chain, see for example Robert (1996), Robert and Casella (2004) and Martin, Mengersen, and Robert (2005). The following quote of Casella, Robert, and Wells (2004) summarizes these views adequately:

> Unfortunately, the practical implementation of this algorithm might run into serious problems because of the phenomenon of the "absorbing component". When only a small number of observations are allocated to a given component j_0, then the following probabilities are quite small:
>
> 1. The probability of allocating new observations to the component j_0.
> 2. The probability of reallocating, to another component, observations already allocated to j_0.
>
> Even though the Gibbs chain $(\mathbf{z}^t, \boldsymbol{\theta}^t)$ is irreducible, the practical setting is one of an almost-absorbing state which is called a trapping state as it may require

> an enormous number of iterations to escape from this state. In extreme cases, the probability of escape is below the minimal precision of the computer and the trapping state is truly absorbing, due to computer "rounding errors". ... ... This is also shown in the lack of proper exploration of the posterior surface, since the Gibbs sampler often exhibits a lack of label switching, that is, the recovery of the invariance of the posterior distribution under permutations of the indices.

To improve mixing, dealing with trapping states and the problem of label switching, alternative MCMC strategies to fit a finite mixture model with k components have been proposed: Metropolis-Hastings or simulated tempering, see for example Celeux, Hurn, and Robert (2000) and Jasra, Holmes, and Stephens (2005). Note that these samplers avoid the use of the latent allocations (2.2). Another alternative, to deal with these problems, is to use a trans-dimensional sampler and then extract the MCMC output for a given k, see Jasra, Holmes, and Stephens (2005). This is our preferred option, and agrees with our purpose: in the next section we build the case for an unknown number of components. However, later on in the thesis we will compare posterior mean estimates calculated with the MCMC output generated via the standard Gibbs sampler against those calculated from the MCMC output of a trans-dimensional sampler. To solve the label switching problem we will use a relabeling algorithm; see Chapter 4.

2.3 Unknown number of components

There are two main alternatives to make inference for the number of components of a finite mixture. We can use reversible jump ideas (Green (1995)) as in Richardson and Green (1997), or a continuous time birth-and-death processes as in Stephens (2000a). Cappé, Robert, and Rydén (2003) investigated the similarity between these methodologies and concluded that the birth-and-death sampler is slower than its reversible jump

counterpart, but its advantage is that it is able to move to unlikely places of the parameter space.

In our case, we will follow reversible jump ideas closely. However, we see reversible jump as a particular case of the product space model discussed by Godsill (2001). From this perspective, the acceptance probability for the reversible jump methodology is deduced directly as a standard Metropolis-Hastings move. The advantage of this alternative deduction is that we can understand better how to derive, use and modify trans-dimensional samplers for non-trivial problems e.g. mixture models. Godsill extended the Carlin and Chib (1995) ideas by introducing a general product space model that comprises many trans-dimensional algorithms, including the reversible jump methodology of Green (1995); see Green (2003). First, a description of the product space model is given and then the acceptance probability for the Metropolis-Hastings step in the product space, is deduced.

2.3.1 Product space model

We suppose that our observations have been generated by a model within a countable collection of candidate models $\{\mathcal{M}_k,\ k\ \in \mathcal{K}\}$ where $\mathcal{K}$ is a set of candidate model indices. Model $\mathcal{M}_k$ has a vector or matrix $\phi^{(k)}$ of unknown parameters. Each parameter $\phi^{(k)}$ has support $\Phi^{(k)}$ and for models with different indices the dimension of the parameters may vary. Our goal would be to choose the best model for the data within all the possible models. The solution given within the Bayesian setting is to calculate the posterior distribution $p(\phi^{(k)}, k|\mathbf{y})$. To this end, we could follow the ideas of Green (1995). In this setting, for a given index k; $(k, \phi^{(k)}) \in \left(\{k\} \times \Phi^{(k)}\right)$ and in general, for a moving model index k,

$$(k, \phi^{(k)}) \in \bigcup_{k \in \mathcal{K}} \left(\{k\} \times \Phi^{(k)}\right). \tag{2.10}$$

Here the idea is to devise a Markov chain with invariant distribution $p(\phi^{(k)}, k|\mathbf{y})$, that traverses the space (2.10), generating proposals $q(k', \phi^{(k')}|k, \phi^{(k)})$ to jump through sub-

spaces of different dimensions, which are then accepted with probability $\alpha_{k,k'}$. To ensure convergence of the chain, to the correct invariant probability distribution, the proposals must satisfy the detailed balance condition, see Appendix A. Once the overall detailed balance is written, the acceptance probability for the reversible jump methodology is worked out. This is rather an obscure procedure. Hence, instead of considering the finite dimensional parameters $\phi^{(k)}$, we consider

$$\phi = (\phi^{(1)}, \phi^{(2)}, \phi^{(3)}, \ldots),$$

so we do not think of jumps between sub-spaces of different dimensions. We define a probability distribution over the entire product space of candidate models and their parameters, so that

$$(k, \phi) \ \in \ \mathcal{K} \times \bigotimes_{k \in \mathcal{K}} \Phi^{(k)}. \tag{2.11}$$

Thus, in the product space model, we change (2.10) for (2.11). The likelihood and the prior structure are defined in a corresponding way as follows; for a particular k the likelihood depends only on the corresponding vector of parameters $\phi^{(k)}$, that is $p(\mathbf{y}|\phi, k) = p(\mathbf{y}|\phi^{(k)}, k)$. The model will be completed by the prior $p(\phi|k)$ and the prior $p(k)$. Then the full posterior distribution of the product space model can be expressed as

$$\begin{aligned} p(\phi, k|\mathbf{y}) &= \frac{p(\mathbf{y}|\phi, k)p(\phi|k)p(k)}{p(\mathbf{y})} \\ &= \frac{p(\mathbf{y}|\phi^{(k)}, k)p(\phi^{(k)}|k)p(\phi^{(-k)}|\phi^{(k)}, k)p(k)}{p(\mathbf{y})} \\ &= p(\phi^{(k)}, k|\mathbf{y})p(\phi^{(-k)}|\phi^{(k)}, k) \end{aligned} \tag{2.12}$$

where $\phi^{(-k)}$ denotes the model parameters in $\Phi^{(j)}$ for all $j \neq k$. The specification of the product space model for a given k is completed by the usual hierarchical structure for the models in $\Phi^{(k)}$ and by $p(\phi^{(-k)}|\phi^{(k)}, k)$ which would be the "priors" or pseudo-priors (this term was first used by Carlin and Chib (1995)) for the parameters that are not used by models in $\Phi^{(k)}$. We could assign any proper distribution to $p(\phi^{(-k)}|\phi^{(k)}, k)$, however,

when reversible jump is derived from the product space model, the pseudo-priors become just a conceptual step. Now, if from (2.12), we integrate out $\phi^{(j)}$ for all $j \neq k$ we obtain $p(\phi^{(k)}, k|\mathbf{y})$, which is the target distribution.

2.3.1.1 Product space Metropolis-Hastings

Godsill (2001) showed that reversible jump is a special case of a Metropolis-Hastings in the product space. To do this, he obtained the reversible jump's acceptance probability to update a Markov chain at a state (k, ϕ) to a new state (k', ϕ') imposing the proposal

$$q(\phi', k'|\phi, k) = q_1(k'|k)q_2(\phi'^{(k')}|\phi^{(k)})p(\phi'^{(-k')}|\phi'^{(k')}, k'), \tag{2.13}$$

where $q_1(k'|k)$ is a proposal to move the model index from k to k' and $q_2(\phi'^{(k')}|\phi^{(k)})$ is a proposal to move the parameters from the model indexed with k to one indexed with k'. Once (2.13) is set and following the product space model (2.12), it is easy to see that (see Section A.4.2 in Appendix A)

$$\begin{aligned}
\alpha_{k,k'}(\phi^{(k)}, \phi'^{(k')}) &= \min\left\{1, \frac{p(\phi', k'|\mathbf{y})q(\phi, k|\phi', k')}{p(\phi, k|\mathbf{y})q(\phi', k'|\phi, k)}\right\} \\
&= \min\left\{1, \frac{p(\phi', k'|\mathbf{y})q_1(k|k')q_2(\phi^{(k)}|\phi'^{(k')})p(\phi^{(-k)}|\phi^{(k)}, k)}{p(\phi, k|\mathbf{y})q_1(k'|k)q_2(\phi'^{(k')}|\phi^{(k)})p(\phi'^{(-k')}|\phi'^{(k')}, k')}\right\} \\
&= \min\left\{1, \frac{p(\phi'^{(k')}, k'|\mathbf{y})q_1(k|k')q_2(\phi^{(k)}|\phi'^{(k')})}{p(\phi^{(k)}, k|\mathbf{y})q_1(k'|k)q_2(\phi'^{(k')}|\phi^{(k)})}\right\}
\end{aligned} \tag{2.14}$$

because in expression (2.14)

$$\begin{aligned}
p(\phi', k'|\mathbf{y}) &= p(\phi^{(k')}, k'|\mathbf{y})p(\phi^{(-k')}|\phi^{(k')}, k') \\
p(\phi, k|\mathbf{y}) &= p(\phi^{(k)}, k|\mathbf{y})p(\phi^{(-k)}|\phi^{(k)}, k)
\end{aligned}$$

hence the therms

$$p(\phi^{(-k)}|\phi^{(k)}, k) \text{ and } p(\phi'^{(-k')}|\phi'^{(k')}, k')$$

are canceled out. But this acceptance probability is rather general. To deduce the most convenient expression for the reversible jump methodology from (2.14), we assume that

the dimension of $\phi^{(k)}$ is n_k and the dimension of $\phi'^{(k')}$ is $n_{k'}$ with $n_{k'} > n_k$. Then, the key idea is to achieve the so called "dimension matching" between $\phi^{(k)}$ and $\phi'^{(k')}$. To this end, we generate a vector or matrix $\mathbf{u}$ with distribution $q_2(\mathbf{u})$ independent of $\phi^{(k)}$ such that the dimension of $(\phi^{(k)}, \mathbf{u})$ is $n_{k'}$ (the idea of Green (1995)). Then, we devise a function T such that

$$T(\phi^{(k)}, \mathbf{u}) = \phi'^{(k')} \quad \text{and} \quad T^{-1}(\phi'^{(k')}) = (\phi^{(k)}, \mathbf{u}). \tag{2.15}$$

Here, to apply the Change of Variable Theorem (CHVT), T must be a bijection and T and T^{-1} must be differentiable.

Then let $q_{2_{\phi^{(k)},\mathbf{u}}}(\cdot|\cdot)$, $q_{2_{\phi^{(k)}}}(\cdot|\cdot)$ denote the joint conditional distributions of $(\phi^{(k)}, \mathbf{u})$, $\phi^{(k)}$, respectively, and $q_{2_{\mathbf{u}}}(\cdot)$ the joint marginal distribution for $\mathbf{u}$. The CHVT can be used to write

$$\begin{aligned}
q_2(\phi'^{(k')}|\phi^{(k)}) &= q_{2_{\phi^{(k)},\mathbf{u}}}(T^{-1}(\phi'^{(k')})|\phi^{(k)}) \left|\frac{\partial T^{-1}(\phi'^{(k')})}{\partial \phi'^{(k')}}\right|, \\
&= q_{2_{\phi^{(k)},\mathbf{u}}}(\phi^{(k)}, \mathbf{u}|\phi^{(k)}) \left|\frac{\partial T^{-1}(\phi'^{(k')})}{\partial \phi'^{(k')}}\right|, \\
&= q_{2_{\phi^{(k)}}}(\phi^{(k)}|\phi^{(k)}) q_{2_{\mathbf{u}}}(\mathbf{u}|\phi^{(k)}) \left|\frac{\partial T^{-1}(\phi'^{(k')})}{\partial \phi'^{(k')}}\right|, \\
&= q_{2_{\mathbf{u}}}(\mathbf{u}) \left|\frac{\partial T^{-1}(\phi'^{(k')})}{\partial \phi'^{(k')}}\right|,
\end{aligned} \tag{2.16}$$

because $\mathbf{u}$ is independent of $\phi^{(k)}$, and $q_{2_{\phi^{(k)}}}(\phi^{(k)}|\phi^{(k)}) = 1$. On the other hand $q_2(\phi^{(k)}|\phi'^{(k')})$ is always 1 because in (2.15) we are assuming that a function such that $\varphi(\phi'^{(k')}) = \phi^{(k)}$ exists.

Then, from (2.16), (2.14) becomes

$$\alpha_{k,k'}(\phi^{(k)}, \phi'^{(k')}) = \min\left\{1, \frac{p(\phi'^{(k')}, k'|\mathbf{y}) q_1(k|k')}{p(\phi^{(k)}, k|\mathbf{y}) q_1(k'|k) q_{2_{\mathbf{u}}}(\mathbf{u})} \left|\frac{\partial T(\phi^{(k)}, \mathbf{u})}{\partial(\phi^{(k)}, \mathbf{u})}\right|\right\} \tag{2.17}$$

because of the Inverse Function Theorem (see for example Rudin (1976)).

Equation (2.17) is usually obtained following reversible jump ideas (see Green (1995), expression (8)). However, the product space formulation is a "standard" Metropolis-Hastings.

2.3.1.2 Product space Metropolis-Hastings for mixtures

For finite mixture models, with an unknown number of components, we need to generate a Markov chain with invariant probability distribution

$$p(\phi^{(k)}, \tau^{(k)}, k|\mathbf{y}) \propto p(\mathbf{y}|\phi^{(k)}, \tau^{(k)}, k)p(\phi^{(k)}, \tau^{(k)}|k)p(k) \tag{2.18}$$

where the $(\tau^{(k)})$ are the discrete latent allocation variables of the model with k components, and the $(\phi^{(k)})$ are the continuous parameters of the model. To obtain the acceptance probability to update the chain from state $(\phi^{(k)}, \tau^{(k)})$ to a new state $(\phi'^{(k')}, \tau'^{(k')})$, there is no need to go back to the product space model, we just rewrite the acceptance probability (2.14) as

$$\alpha_{k,k'}((\phi^{(k)}, \tau^{(k)}), (\phi'^{(k')}, \tau'^{(k')})) = \min\left\{1, \frac{\pi_{k'}}{\pi_k}\right\} \tag{2.19}$$

with

$$\frac{\pi_{k'}}{\pi_k} = \frac{p(\phi'^{(k')}, \tau'^{(k')}, k'|\mathbf{y})q_1(k|k')q_2(\phi^{(k)}, \tau^{(k)}|\phi'^{(k')}, \tau'^{(k')})}{p(\phi^{(k)}, \tau^{(k)}, k|\mathbf{y})q_1(k'|k)q_2(\phi'^{(k')}, \tau'^{(k')}|\phi^{(k)}, \tau^{(k)})}.$$

In our case, the proposals for the continuous variables of the model are going to be independent of the proposed allocations, but for the allocations, the proposal does depend on the proposed continuous parameters. Thus,

$$q_2(\phi^{(k)}, \tau^{(k)}|\phi'^{(k')}, \tau'^{(k')}) = q_2(\phi^{(k)}|\phi'^{(k')})q_2(\tau^{(k)}|\phi'^{(k')}, \tau'^{(k')})$$

so under this formulation we can use again (2.16), to obtain an applied version of (2.19),

$$\frac{\pi_{k'}}{\pi_k} = \frac{p(\phi'^{(k')}, \tau'^{(k')}, k'|\mathbf{y})q_1(k|k')}{p(\phi^{(k)}, \tau^{(k)}, k|\mathbf{y})q_1(k'|k)q_{2_\mathbf{u}}(\mathbf{u})} \left|\frac{\partial T(\phi^{(k)}, \mathbf{u})}{\partial(\phi^{(k)}, \mathbf{u})}\right| \frac{q_2(\tau^{(k)}|\phi'^{(k')}, \tau'^{(k')})}{q_2(\tau'^{(k')}|\phi^{(k)}, \tau^{(k)})}.$$

Under the product space model, (2.19) is a straightforward consequence of (2.17).

2.3.2 Univariate normal mixtures (Richardson and Green (1997))

We will use the mixture of univariate normal distributions as our benchmark model, see Section 2.2.2. First, let us identify the variables according to (2.19):

$$\begin{aligned} \phi^{(k)} &= \left(\mathbf{w}^{(k)}, \boldsymbol{\mu}^{(k)}, \boldsymbol{\sigma}^{2(k)}\right) \\ \tau^{(k)} &= \mathbf{z}^{(k)} \end{aligned}$$

where $\mathbf{w}^{(k)} = \{w_j\}_{j=1}^k$, $\boldsymbol{\mu}^{(k)} = \{\mu_j\}_{j=1}^k$ and $\boldsymbol{\sigma}^{2(k)} = \{\sigma^2{}_j\}_{j=1}^k$. Note that β does not depend on the number of components of the mixture.

The strategy, transformations and Jacobian to calculate (2.19) can be found in Richardson and Green (1997). But as we will use some of their ideas for our own model, we give a detailed description.

The Markov chain is constructed via a hybrid strategy (see Appendix A, Section A.4.4), we will have the following moves:

1. For a fixed k, a Gibbs kernel is used; this move was described in Section 2.2.2 (ordering the means).

2. Split-combine move; this move is a mixture of two Metropolis-Hastings kernels, both attempting to change the number of components of the mixture.

3. Birth and death of empty components; this move again is a mixture of two Metropolis-Hastings kernels, both attempting to change the number of components of the mixture.

4. Once the birth and death move has been attempted we return to the first step, hence forming cycle. This is repeated until the sampler has converged.

Split-combine: in the split, one component, let us say component j, is chosen randomly from $\{1, \ldots, k\}$, and split into two consecutive components, let us say components $j_1 < j_2$. The aim is to preserve the order of the means. In the combine, this is exactly

the "inverse move", two consecutive components, $j_1 < j_2$, are selected randomly from $\{1, \ldots, k+1\}$ and combined into one component, j. The number of components is always within the finite set $\{1, 2, \ldots, k_{max}\}$. The integer k_{max} depends on the prior for k, e.g. the prior for k could be discrete uniform over the range $\{1, 2, \ldots, k_{max}\}$ or a truncated Poisson within the same range. Let b_k and d_k be the probabilities of proposing the split or combine move respectively. These are usually taken as $b_k = d_k = \frac{1}{2}$ for $k = 2, 3, \ldots, k_{max} - 1$ and $b_1 = d_{k_{max}} = 1$ where $b_k + d_k = 1$ for $k = 1, \ldots, k_{max}$.

The first step is to make a random choice between to attempt the move from a mixture with k components to one with $k+1$, or from a mixture with k components to one with $k-1$ components. This choice is made with probabilities b_k and d_k as above. For this description we assume that we are attempting to move a mixture of k components to one with $k+1$ components, thus in (2.19) we take $k' = k+1$. Under this assumption, the proposals $q_1(k+1|k)$ and $q_1(k|k+1)$ are already set and given by

- $q_1(k+1|k) = b_k/k$: with a mixture of k components, the probability of proposing the split move, times the probability of choosing the component j.
- $q_1(k|k+1) = d_{k+1}/k$: with a mixture of $k+1$ components, the probability of proposing the combine move, times the probability of choosing two consecutive components $j_1 < j_2$.

Then, the ratio to update the model index is given by

$$\frac{q_1(k|k+1)}{q_1(k+1|k)} = \frac{d_{k+1}}{b_k} \quad \text{for } k = 1, 2, \ldots, k_{max}. \tag{2.20}$$

We need to devise proposals for the continuous and discrete variables, the proposals for the continuous variables are devised first.

For the split, generate $\mathbf{u} = (u_1, u_2, u_3)$, via

$$u_1 \sim \text{beta}(2,2), \quad u_2 \sim \text{beta}(2,2), \quad u_3 \sim \text{beta}(1,1)$$

and then calculate

$$\begin{aligned} w_{j_1} = u_1 w_j \quad &\leftrightarrow \quad w_{j_2} = (1-u_1)w_j \\ \mu_{j_1} = \mu_j - \frac{u_2\sqrt{\sigma_j^2 u_1(1-u_1)}}{u_1} \quad &\leftrightarrow \quad \mu_{j_2} = \mu_j + \frac{u_2\sqrt{\sigma_j^2 u_1(1-u_1)}}{1-u_1} \\ {\sigma^2}_{j_1} = u_3(1-u_2^2){\sigma^2}_j\frac{1}{u_1} \quad &\leftrightarrow \quad {\sigma^2}_{j_2} = (1-u_3)(1-u_2^2){\sigma^2}_j\frac{1}{1-u_1}. \end{aligned}$$

The location parameters must satisfy $\mu_{j-1} < \mu_{j_1}$ and $\mu_{j_2} < \mu_{j+1}$ ($\mu_0 = -\infty$ and $\mu_{k+1} = \infty$) if not, the move is rejected immediately.

The combine is derived directly, we work out the variables from the split to obtain

$$\begin{aligned} w_j = w_{j_1} + w_{j_2} \quad &\leftrightarrow \quad u_1 = \frac{w_{j_1}}{w_{j_1} + w_{j_2}} \\ \mu_j = (w_{j_1}\mu_{j_1} + w_{j_2}\mu_{j_2})/w_j \quad &\leftrightarrow \quad u_2 = \frac{\sqrt{w_{j_1}w_{j_2}}(\mu_{j_2} - \mu_{j_1})}{\sqrt{w_{j_1}w_{j_2}(\mu_{j_2}-\mu_{j_1})^2 + w_j(w_{j_1}\sigma_{j_1}^2 + w_{j_2}\sigma_{j_2}^2)}} \\ \sigma_j^2 = \frac{w_{j_1}w_{j_2}(\mu_{j_2}-\mu_{j_1})^2}{w_j^2 u_2^2} \quad &\leftrightarrow \quad u_3 = \frac{w_{j_1}\sigma_{j_1}^2}{w_{j_1}\sigma_{j_1}^2 + w_{j_2}\sigma_{j_2}^2}. \end{aligned}$$

Remark 2.3. These are the transformations mentioned in (2.15), the proposals for the split are generated by T and for the combine by T^{-1}. For this particular case (i.e. univariate normal mixtures), the idea is to try to match the first and second moments of the new components to those of the two that it replaces. This idea just gives an insight for the combine, but obtaining the transformations for the variables $\mathbf{u} = (u_1, u_2, u_3)$ is completely a matter of trial and error. Defining the transformation T for nontrivial problems can be really challenging.

Remark 2.4. These transformations have been designed for the variance, σ_j^2, while the model was devised for the precision, τ_j. Hence, when calculating (2.19) the change of variable $\sigma_j^2 = \frac{1}{\tau_j} \sim \text{Inv-Ga}(\sigma_j^2|\alpha, \beta)$ must be considered.

It is now possible to calculate the Jacobian in expression (2.19), this is given by

$$\left|\frac{\partial T(\phi^{(k)}, \mathbf{u})}{\partial(\phi^{(k)}, \mathbf{u})}\right| = \frac{w_j(1-u^2)\sigma_j^3}{(u_1(1-u_1))^{\frac{3}{2}}}. \tag{2.21}$$

Expression (2.21) is not the same as the one displayed in Richardson and Green (1997), but is easy to show that both expressions are equivalent.

The proposals for the (discrete) allocation variables are as follows: generate a vector $\mathbf{s} = \{s_i\}_{i=1}^{n}$ such that $s_i = z_i$ for $i = 1, \ldots, n$. For the split, first, all the observations such that $s_i = j^*$ with j^* larger than j are re-allocated to $s_i = j^* + 1$. Second, the observations such that $s_i = j$ must be re-allocated randomly to component j_1 or component j_2, with the following probabilities

$$p_{i,l} = \frac{w_{j_l} N(y_i|\mu_{j_l}, \sigma^2_{j_l})}{w_{j_1} N(y_i|\mu_{j_1}, \sigma^2_{j_1}) + w_{j_2} N(y_i|\mu_{j_2}, \sigma^2_{j_2})}, \tag{2.22}$$

for $l = 1, 2$ and for all i such that $z_i = j$.

With $\mathbf{s}$ already re-allocated, and using the probabilities (2.22), we calculate the probability for the proposal of the discrete variables, this is given by

$$p(\tau^{(k+1)}|\phi^{(k)}, \tau^{(k)}) = \prod_{\{i:z_i=j\}} p_{i,s_i}. \tag{2.23}$$

For the combine; we re-allocate the $s_i = j^*$ bigger than or equal to j_2 into $j^* - 1$. This is a deterministic move, thus the probability for the proposal of the discrete variables in the combine is

$$p(\tau^{(k)}|\phi^{(k+1)}, \tau^{(k+1)}) = 1. \tag{2.24}$$

Then, in (2.19) we substitute the values of (2.23) and (2.24). Note that Richardson and Green named (2.23) as $\mathbf{P}_{alloc}$. From the product space derivation it easy to see why and from where $\mathbf{P}_{alloc}$ appears, on the other hand, from the reversible jump perspective this is rather obscure.

To finish the specification of (2.19) we only need to calculate the ratio

$$\begin{aligned}\frac{p(\phi'^{(k')},\tau'^{(k')},k'|\mathbf{y})}{p(\phi^{(k)},\tau^{(k)},k|\mathbf{y})q_{2\mathbf{u}}(\mathbf{u})} &= \frac{\prod_{i\in S_1\bigcup S_2}\mathrm{N}(y_i|\mu_{s_i},\sigma^2_{s_i})}{\prod_{i\in Z}\mathrm{N}(y_i|\mu_j,\sigma^2_j)}\frac{p(k+1)}{p(k)}\frac{w_{j_1}^{\delta-1+n_{j_1}}w_{j_2}^{\delta-1+n_{j_2}}}{w_j^{\delta-1+n_j}B(\delta,k\delta)}\\ &\times (k+1)\sqrt{\frac{\kappa}{2\pi}}e^{-\frac{\kappa}{2}\left[(\mu_{j_1}-\mu_0)^2+(\mu_{j_2}-\mu_0)^2-(\mu_j-\mu_0)^2\right]}\\ &\times \frac{\beta^\alpha}{\Gamma(\alpha)}\left(\frac{\sigma^2_{j_1}\sigma^2_{j_2}}{\sigma^2_j}\right)^{-\alpha-1}e^{-\beta(\sigma_{j_1}^{-2}+\sigma_{j_2}^{-2}-\sigma_j^{-2})}\\ &\times \{g_{2,2}(u_1)g_{2,2}(u_2)g_{1,1}(u_3)\}^{-1} \qquad (2.25)\end{aligned}$$

with $g_{a,b}(\cdot)$ the beta density function with parameters a, b and where $S_1 = \{i : s_i = j_1\}$, $S_2 = \{s_i = j_2\}$ and $Z = \{i : z_i = j\}$, hence $n_{j_1} = \#S_1$, $n_{j_2} = \#S_2$ and $n_j = \#Z$.

To attempt the move from a mixture of k components to one of $k-1$ components we use the acceptance probability

$$\alpha_{k,k-1}((\phi^{(k)},\tau^{(k)}),(\phi'^{(k-1)},\tau'^{(k-1)})) = \min\left\{1,\left(\frac{\pi_k}{\pi_{k-1}}\right)^{-1}\right\}, \qquad (2.26)$$

instead of (2.19), thus the previous description is still valid.

Birth and death of empty components: For the birth, a new empty component is generated. For the death, a random choice is made between any existing empty components and the chosen component is deleted. An empty component is a component who has no observations assigned to it: the number of empty components, for a mixture of k components, will be denoted as k_0.

As with the split and combine move, we need to make a random choice between proposing the move from a mixture with k components to one with $k+1$ or to a mixture with $k-1$ components. This choice is made with probabilities b_k and d_k as before. We again assume that we are attempting to move a mixture of k components to one with $k+1$ components. Under this assumption, the proposals $q_1(k+1|k)$ and $q_1(k|k+1)$ are given by

- $q_1(k+1|k) = b_k$: with a mixture of k components, the probability of to attempt the birth move.

- $q_1(k|k+1) = d_{k+1}/(k_0+1)$: with a mixture of $k+1$ components, the probability of to attempt the death move, times the probability of choosing one of the k_0+1 empty components.

Then, the ratio to update the model index is given by

$$\frac{q_1(k|k+1)}{q_1(k+1|k)} = \frac{d_{k+1}}{b_k(k_0+1)} \quad \text{for } k = 1, 2, \ldots, k_{max}.$$

Since we are generating or deleting empty components, the allocation variables must not be modified, then the ratio of proposals for the discrete variables in (2.19) is 1. Hence, just proposals for the continuous variables are needed.

For the birth, a weight and parameters for the proposed new component are drawn using

$$w_{j^*} \sim \text{beta}(1, k), \quad \mu_{j^*} \sim \text{N}(\mu_0, \kappa^{-1}), \quad \sigma^2_{j^*} \sim \text{Inv-Ga}(\alpha, \beta), \tag{2.27}$$

and the existing weights are rescaled, so that all the weights sum to 1, i.e.

$$w_{j'} = w_j(1 - w_{j^*}). \tag{2.28}$$

For the death, between the existing empty components, an empty component is randomly chosen and deleted. The remaining weights are rescaled to sum to 1, i.e. if j^* is the chosen empty component $w_{j'} = w_j/(1 - w_{j^*})$.

Remark 2.5. In the birth, the new empty component must be accommodated so that the order of the means is satisfied.

The latent allocations remain unchanged and the proposals (2.27) play the role of the independent variable $\mathbf{u} = (w_{j^*}, \mu_{j^*}, \sigma^2_{j^*})$, in (2.19), these two facts lead to a simplification of the ratio:

$$\frac{p(\phi'^{(k')}, \tau'^{(k')}, k'|\mathbf{y})}{p(\phi^{(k)}, \tau^{(k)}, k|\mathbf{y})q_{2\mathbf{u}}(\mathbf{u})} = \frac{p(k+1)}{p(k)}(k+1)\frac{w_{j^*}^{\delta-1}(1-w_{j^*})^{n+k\delta-k}}{B(k\delta, \delta)g_{1,k}(w_{j^*})}.$$

Finally, to calculate the Jacobian, we need to consider (2.28) and $\sum_j w_j = 1$, thus

$$\left|\frac{\partial T(\phi^{(k)}, \mathbf{u})}{\partial(\phi^{(k)}, \mathbf{u})}\right| = (1 - w_j^*)^{k-1},$$

see Richardson and Green (1998).

To attempt the move from a mixture of k components to one of $k-1$, we use again (2.26). This completes the specification of the trans-dimensional sampler to make inference for the number of components for a finite mixture of univariate normals as in Richardson and Green (1997). To make inference for the number of components for mixtures of Poisson distributions or multivariate normal distributions, see Viallefont, Richardson, and Green (2002) and Dellaportas and Papageorgiou (2006), respectively.

2.3.2.1 Prior specification

Later on in the thesis we will perform several comparisons using the mixtures of univariate normal distributions as benchmark model, and we will follow Richardson and Green (1997) ideas to fix the unspecified constants of the priors. Under this setting, it is assumed that we do not have strong prior information on the mixture parameters. The aim is to work under weak informative priors, and base the unspecified constants only on the range of the data: the range of the data is given by $R = y_{(n)} - y_{(1)}$. Although k is considered to be unknown, our hierarchical model (2.6) admits the representation $p(k, \gamma, \boldsymbol{\theta}, \mathbf{w}, \mathbf{z}, \mathbf{y}) \propto p(k)p(\gamma, \boldsymbol{\theta}, \mathbf{w}, \mathbf{z}|k, \mathbf{y})$. Thus, the prior structure can be set in terms of the known k model:

$$\mu_0 = y_{(1)} + \frac{R}{2}, \tag{2.29}$$

$$\kappa = \frac{1}{R^2}, \tag{2.30}$$

$$\alpha = 2, \tag{2.31}$$

$$g = 0.2, \tag{2.32}$$

$$h = \frac{100g}{\alpha R^2}, \tag{2.33}$$

$$\delta = 1. \tag{2.34}$$

The prior over μ_j, (2.29) and (2.30), reflects vague knowledge about the location of the means, and the choice for the weights, (2.34), gives a uniform prior over the space $w_1 + \cdots + w_k = 1$. However, the prior over $\tau_j = \sigma_j^{-2}$, (2.31)-(2.33), is more complicated than this, as expressed in the remark 2.1. Here, following Richardson and Green (1997),

we give a more comprehensive explanation.

First note that

$$p(\sigma_j^{-2}) = \text{Ga}(\alpha, \beta) \Rightarrow \mathbb{E}\left(\sigma_j^{-2}\right) = \frac{\alpha}{\beta}, \tag{2.35}$$

$$p(\beta) = \text{Ga}(g, h) \Rightarrow \mathbb{E}(\beta) = \frac{g}{h}, \tag{2.36}$$

and the aim is to make the posterior of k less sensitive to the choice of β. Thus (2.36) was introduced. Now, to devise the default prior, using (2.35) and (2.36), we relate

$$\sigma_j \cong \sqrt{\frac{\beta}{\alpha}} \cong \sqrt{\frac{g}{h\alpha}},$$

and finally the variance is "connected" with the range of the data: $\sqrt{g/(h\alpha)} = pR$, where $(1/20 \leq p \leq 1/5)$. After a sensitivity study on the posterior of k, see pp. 747-748, they chose $p = 1/10$, $\alpha = 2$ and $g = 0.2$. Leading to (2.33), where $\alpha > 1 > g$ expresses the belief that the variances are similar, without being informative about its absolute size, see p. 735.

2.3.2.2 Convergence

We described a hybrid strategy, this is just a composition of several Markov kernels, see Appendix A, Section A.4.4. For a given k, the space state of the chain is updated via a Gibbs kernel, and to move k there are two mixtures of Metropolis-Hastings kernels. Each kernel with invariant probability distribution $p(\phi^{(k)}, \tau^{(k)}, k|\mathbf{y})$, see (2.18). This guarantees the converge of the Markov chain to the correct invariant probability distribution.

Remark 2.6. The Gibbs sampler kernel is not needed: the two mixtures of Metropolis-Hastings kernels alone guarantee the convergence of the chain to the correct posterior distribution, this has been noted before by Cappé, Robert, and Rydén (2003). Indeed, theoretically, only one mixture of Metropolis-Hastings will be needed to ensure convergence to the invariant probability distribution.

It is difficult to asses Harris recurrence (see Section A.3), but irreducibility is easily established: k can traverse the set $\{1, \ldots, k_{max}\}$, by moving in steps of one at a time and for a given k, all the parameters are updated via a Gibbs sampler, which is irreducible. To establish aperiodicity, we need to show that if at time t the chain is at the state $(k, \theta^{(k)}, \tau^{(k)})$, and we define an arbitrarily small neighborhood of it, then at time $t+1$ there is positive probability that the chain lies in that same neighborhood. But this is true, since there is positive probability to reject the two moves that alter the number of components of the mixture, and given k, the Gibbs sampler is irreducible.

Remark 2.7. Mixtures of Metropolis-Hastings combined with the Gibbs sampler for a fixed k is an attempt to improve mixing.

Thus, to approximate relevant quantities by averages of sample paths of the chain, we still can use the relaxed version of the Ergodic Theorem in Appendix A. For example, we can approximate the posterior distribution for the number of components, i.e.

$$\begin{aligned} p(k=j|\mathbf{y}) &= \mathbb{E}\left(\mathbb{1}\{k=j\}|\mathbf{y}\right) \\ &\approx \frac{1}{N}\sum_{t=1}^{N}\mathbb{1}\{k^t=j\} \quad \text{(for } N \text{ large enough)}, \end{aligned} \tag{2.37}$$

for all $j \in 1, \ldots, k_{max}$. To generate a predictive density estimate, we simply average over different values of k, i.e.

$$p(y^*|\mathbf{y}) \approx \frac{1}{N}\sum_{t=1}^{N}\left\{\sum_{j=1}^{k^t} w_j^t f(y^*|\theta_j^t)\right\} \text{(for } N \text{ large enough)},$$

where $(k^t, \mathbf{w}^t, \boldsymbol{\theta}^t)$ for $t = 1, \ldots, N$ are generated via a hybrid strategy (N is the number of iterations after a burn in period).

2.4 Additional comments

The analysis for the fixed k model can be carried out simply extracting the output for a relevant k, from the trans-dimensional sampler. On this line, Richardson and Green

(1997) and Stephens (2000a) observed that the moving k sampler can be used to improve the analysis of the fixed k model. This is because the trans-dimensional sampler is able to move around the modes of the posterior distribution of the fixed k model, via models of lower or higher dimensions. Thus visiting the posterior modes of the fixed k model, in a more efficient manner, i.e. taking less iterations to escape trapping states and in general mixing better within k.

To ensure convergence of the trans-dimensional sampler we need to generate larger number of iterations than with its fixed dimension version. This is because the state space of the chain for the model with an unknown k is larger. For the trans-dimensional sampler, for mixtures of univariate normals, we usually generate 1,000,000 of iterations and discard the first 200,000 as a burn in period. While for the Gibbs sampler 60,000 iterations are generated and then the first 30,000 are taken as a burn in period.

Chapter 3

MCMC Methods for Infinite Mixtures

This Chapter reviews MCMC methods for fitting Bayesian nonparametric mixture models that are based on the Dirichlet process. In general, nonparametric methods are more complicated to understand, construct and use than their parametric counterparts so our aim in this Chapter is to motivate the ideas rather than present a detailed technical and comprehensive summary of results. We start with an outline of the Dirichlet process.

3.1 The Dirichlet process

The Dirichlet process can be motivated from its closest parametric relative: the multinomial model.

3.1.1 The multinomial model

The multinomial model specifies an arbitrary probability over the space of partitions of n objects into k clusters, and can be derived by defining an arbitrary distribution over the k clusters and then grouping the objects according to this distribution. Concisely, let $z_i \in \{1, \ldots, k\}$ be a discrete random variable with probabilities $p(z_i = j|\mathbf{w}) = w_j$,

where $\mathbf{w} = \{w_j\}_{j=1}^k$, $w_j \geq 0$ for $j = 1, \ldots, k$ and $\sum_{j=1}^k w_j = 1$. Thus, z_i is a variable that indicates in which of the k clusters the object i belongs. For an i. i. d. sample $\mathbf{z} = \{z_i\}_{i=1}^n$, we have that

$$p(\mathbf{z}|\mathbf{w}) \propto \prod_{j=1}^k {w_j}^{n_j}, \tag{3.1}$$

where $n_j = \#\{i : z_i = j\}$ and $n = \sum_{j=1}^k n_j$ (hence $n_j \in \{0, 1, 2, \ldots\}$).

To obtain the proportionality constant we need to account for all possible permutations in which the n elements of $\mathbf{z}$ can be assigned to k different clusters of sizes $\mathbf{n} = \{n_j\}_{j=1}^k$; hence

$$p(\mathbf{z}|\mathbf{w}) = \binom{n}{n_1 \cdots n_k} \prod_{j=1}^k {w_j}^{n_j}, \tag{3.2}$$

where (3.2) is the multinomial distribution and the proportionality constant is known as the multinomial coefficient.

Under the grouping operation the objects can be reduced to counts over the different clusters and the distribution of counts $\mathbf{n}$, over the k clusters, is again (3.2). Thus the random partitions $\mathbf{z}$ and the counts $\mathbf{n}$ follow the same distribution.

The multinomial distribution is a generalization of the binomial distribution and it is very useful to model experiments that can result in one of k possible outcomes. The most convenient prior for $\mathbf{w}$ is a conjugate prior, which in this case is the Dirichlet distribution ($\mathrm{Dir}(\mathbf{w}|\alpha_1, \ldots, \alpha_k)$):

$$p(\mathbf{w}|\alpha_1, \ldots, \alpha_k) = \frac{\Gamma(\sum_{j=1}^k \alpha_j)}{\prod_{j=1}^k \Gamma(\alpha_j)} \prod_{j=1}^k w_j^{\alpha_j - 1}. \tag{3.3}$$

Observe that the mean of each weight is given by

$$\mathbb{E}(w_j) = \frac{\alpha_j}{\sum_{j=1}^k \alpha_j}, \tag{3.4}$$

and also note the beta distribution is a Dirichlet distribution when $k = 2$ (the beta distribution is a conjugate prior for the binomial distribution).

With this choice of prior, the posterior for the weights is given by

$$\begin{aligned} p(\mathbf{w}|\mathbf{z}) &\propto p(\mathbf{z}|\mathbf{w})p(\mathbf{w}|\alpha_1,\ldots,\alpha_k) \propto \prod_{j=1}^{k} w_j^{\alpha_j+n_j-1} \\ &\propto p(\mathbf{w}|\alpha_1+n_1,\ldots,\alpha_k+n_k) \\ &\Rightarrow \mathbf{w}|\mathbf{z} \sim \mathrm{Dir}(\mathbf{w}|\alpha_1+n_1,\ldots,\alpha_k+n_k), \end{aligned} \tag{3.5}$$

and the predictive for z_{n+1} is easily calculated, if we assume that z_{n+1} and $\mathbf{z}$ are conditionally independent given $\mathbf{w}$. Hence

$$\begin{aligned} p(z_{n+1}=j|\mathbf{z}) &= \int p(z_{n+1}=j|\mathbf{w})p(\mathbf{w}|\mathbf{z})\mathrm{d}\mathbf{w} \\ &= \int w_j p(\mathbf{w}|\mathbf{z})\mathrm{d}\mathbf{w} \\ &= \mathbb{E}(w_j|\mathbf{z}) \\ &= \frac{\alpha_j+n_j}{n+\sum_{j=1}^{k}\alpha_j} \end{aligned} \tag{3.6}$$

where (3.6) is consequence of (3.5) and (3.4).

The latent variables (2.3), of the finite mixture model, have the same meaning and are indeed multinomial random variables. Also, note that expression (3.5) was used to update the weights of the finite mixture model via a Gibbs sampler.

Remark 3.1. Since $\mathbf{w}$ is an arbitrary probability distribution over $\{1,\ldots,k\}$, then we can say that the Dirichlet distribution is a convenient prior distribution over the set of finite probability distributions. With this in mind, Ferguson (1973) introduced the Dirichlet process, $\mathscr{P}$, which is a prior distribution over the space of probability distributions: if G is an arbitrary distribution over some very general sample space (it does not have to be finite), the Dirichlet process is used as a convenient prior over G ($G \sim \mathscr{P}$), i.e. it is a conjugate prior and has large support over the space of distribution functions.

3.1.2 Definition and properties

Formally, the Dirichlet process, $\mathscr{P}$, is defined via finite-dimensional Dirichlet distributions.

Definition 3.1.1. Let G_0 be a distribution over Ω and c be a positive real number. We say that $G \sim \mathscr{P}$ where $\mathscr{P}$ is the Dirichlet process measure on $(\Omega, \mathscr{C})$ with base distribution G_0 and concentration parameter c if

$$(G(A_1), \ldots, G(A_m)) \sim \mathrm{Dir}(cG_0(A_1), \ldots, cG_0(A_m)) \tag{3.7}$$

for every measurable partition $A_1, \ldots, A_m$ of Ω.

Remark 3.2. We say that $A_1, \ldots, A_m$ is a measurable partition of Ω if $A_l \subset \mathscr{C}$ for all $l = 1, \ldots, m$, $A_l \bigcap A_h = \emptyset$ for $l \neq h$ and $\bigcup_{l=1}^{m} A_l = \Omega$. Note that if $A \subset \mathscr{C}$, then A, A^c is a measurable partition of Ω.

There are three ways of to prove the existence of the Dirichlet process measure, the first one is via Kolmogorov's Consistency Theorem as in Ferguson (1973). The second one is via de Finetti's Theorem, exchangeability arguments and the Pólya urn scheme as in Blackwell and MacQueen (1973), and the third one is via a constructive definition as in Sethuraman (1994). The constructive argument, also known as the stick-breaking construction, is regarded as the most direct and general proof.

Remark 3.3. If $G \sim \mathscr{P}$ where $\mathscr{P}$ the Dirichlet Process measure with base distribution G_0 and concentration parameter c, then for ease of notation (in most cases) we will avoid the reference to the probability measure simply writing $G \sim \mathrm{DP}(c, G_0)$.

Remark 3.4. A key fact about the Dirichlet process is that it is almost surely a discrete measure, Blackwell (1973), i.e. if $G \sim \mathrm{DP}(c, G_0)$, then G is a discrete distribution.

Remark 3.5. Given that only discrete distributions G can be sampled from a Dirichlet process $\mathrm{DP}(c, G_0)$ it is surprising to observe that its support is quite big, indeed

$$\mathrm{supp}(\mathrm{DP}) = \{G : \mathrm{supp}(G) \subset \mathrm{supp}(G_0)\}.$$

For example, if the support of G_0 is all the real line (take the normal distribution) then every probability measure is in the support of the Dirichlet process, see Ghosal (2010).

With (3.7) we can easily calculate the mean and variance of the Dirichlet process: let $G \sim \mathrm{DP}(c, G_0)$ and $A \subset \mathscr{C}$, since $(G(A), G(A^c)) \sim \mathrm{Be}(cG_0(A), c(1 - G_0(A)))$, thus

$$\mathbb{E}(G(A)) = G_0(A) \quad \text{and} \quad \mathrm{var}(G(A)) = \frac{G_0(A)(1 - G_0(A))}{c + 1}. \tag{3.8}$$

Then the larger c is, the smaller the variance, and the Dirichlet process will concentrate more mass around its mean.

Remark 3.6. If $G \sim \mathrm{DP}(c, G_0)$, thus G is a discrete random distribution and we can in turn draw samples from G itself.

3.1.3 Posterior and predictive distributions

Let G be some distribution and let $\theta_1, \ldots, \theta_n$ be an i. i. d. sequence from G, we can then assign a Dirichlet process prior over G (the support of the Dirichlet process is quite big, despite being a discrete distribution) and calculate the posterior distribution of G. This is a Dirichlet process again:

$$[G|\theta_1, \ldots, \theta_n] \sim \mathrm{DP}\left(c + n, \frac{c}{c + n} G_0 + \frac{n}{c + n} F_n\right), \tag{3.9}$$

where $F_n = \dfrac{1}{n} \displaystyle\sum_{i=1}^{n} \delta_{\theta_i}$ is the empirical distribution function.

From the definition of the Dirichlet process (3.9) indicates that for any measurable partition $A_1, \ldots, A_m$ of Ω

$$[G(A_1), \ldots, G(A_m)|\theta_1, \ldots, \theta_n] \sim \mathrm{Dir}(cG_0(A_1) + n_1, \ldots, cG_0(A_m) + n_m), \tag{3.10}$$

where $n_l = \displaystyle\sum_{i=1}^{n} \delta_{\theta_i}(A_l)$ denotes the number of (θ_i) in A_l.

To see that (3.10) is true, without going into technical details, simply note the conjugacy between the Dirichlet and the multinomial distribution (3.5).

From (3.8), the posterior mean is given by

$$\mathbb{E}(G|\theta_1,\ldots,\theta_n) = \frac{c}{c+n}G_0 + \frac{n}{c+n}F_n, \tag{3.11}$$

and thus the posterior mean is a mixture between the prior mean and the empirical distribution. For a fixed n and large values of c, the posterior mean gives more weight to G_0. On the other hand, for a fixed c and larger values of n the empirical distribution will have more weight.

To finish this Section we consider the predictive distribution of θ_{n+1}, conditional on $\theta_1,\ldots,\theta_n$. An important property to calculate this is that $[\theta_{n+1}|G,\theta_1,\ldots,\theta_n] \sim G$, then for some $A \in \mathscr{C}$ and integrating out G we have

$$\begin{aligned}\mathscr{P}(\theta_{n+1}\in A|\theta_1,\ldots,\theta_n) &= \mathbb{E}(G(A)|\theta_1,\ldots,\theta_n)\\ &= \frac{c}{c+n}G_0(A) + \frac{n}{c+n}F_n(A).\end{aligned}$$

Note that if in (3.6) we take $k=2, \alpha_1 = cG_0(A)$ and $\alpha_2 = c(1-G_0(A))$ we obtain the same result, in that case $\mathbf{w}$ would be playing the role of G.

The sequence of predictive distributions (3.12) for $\theta_1,\theta_2,\theta_3,\ldots$ is called the Pólya or Blackwell and MacQueen urn scheme.

$$\mathscr{P}(\theta_{n+1}|\theta_1,\ldots,\theta_n) = \frac{c}{c+n}G_0 + \frac{n}{c+n}F_n. \tag{3.12}$$

3.1.4 Clustering

If $G \sim \mathrm{DP}(c,G_0)$, then the discrete distribution G generates ties in the observations, and this is extremely useful in clustering applications. In a sample $\theta_1,\ldots,\theta_n$ we may only have k distinct values $\varphi_1,\ldots,\varphi_k$ with $k \leq n$. Thus k represents the number of clusters, and the different values define a clustering structure over $\theta_1,\ldots,\theta_n$. With this idea, letting $n_j = \#\{i : \theta_i = \varphi_j\}$, for $j=1,\ldots,k$, it is possible to rewrite (3.12), as

$$\mathscr{P}(\theta_{n+1}|\theta_1,\ldots,\theta_n) = \frac{c}{c+n}G_0 + \frac{1}{c+n}\sum_{j=1}^{k} n_j\delta_{\varphi_j}, \tag{3.13}$$

Remark 3.7. This process defines discrete latent variables: take $z_i = j$ iff $\theta_i = \varphi_j$, for $i = 1, \ldots, n$, then z_i indicates to which cluster each θ_i belongs. Under this setting the number of clusters, k, is the number of distinct values in $\mathbf{z} = \{z_i\}_{i=1}^n$ ($n_j = \#\{i : z_i = j\}$). Thus, from equation (3.13) we have that

$$p(z_{n+1} = l | c, n, z_1, \ldots, z_n) = \frac{c}{c+n}\delta_{k+1}(l) + \frac{1}{c+n}\sum_{j=1}^{k} n_j \delta_j(l). \tag{3.14}$$

The discrete distribution (3.14) says that the number of clusters can change between realizations of the experiment.

Remark 3.8. Once z_{n+1} has been drawn, and given $\varphi_1, \ldots, \varphi_k$, it is easy to work out θ_{n+1}. If $z_{n+1} \in \{1, \ldots, k\}$, then $\theta_{n+1} = \varphi_{z_{n+1}}$ and if $z_{n+1} \notin \{1, \ldots, k\}$, then we draw a new value via $\theta_{n+1} \sim G_0$. The same is true with θ_{n+1}. If say $\theta_{n+1} = \varphi_j$, then $z_{n+1} = j$, and if $\theta_{n+1} \neq \varphi_j$, for $j = 1, \ldots, k$, then $z_{n+1} = k + 1$ (θ_{n+1} must have been drawn via $\theta_{n+1} \sim G_0$).

The $\mathbf{z}$ induce random partitions over the set $\{1, \ldots, n\}$ (similarly to those induced in the multinomial model). These random partitions define groups or clusters among the observations, and we could try to proceed as in multinomial model. Studying the distribution of the random partitions of n objects into k groups, but this will be far more complex than those of the multinomial model. In that case the number of clusters is fixed, through different realizations of the experiment, instead for the Dirichlet process k can change. This is beyond the scope of this review, however, it is interesting to note that all the properties of the Dirichlet process can be deduced studying these random partitions, see for example Pitman (2006).

From (3.14) we can calculate the expected number of clusters among $n + 1$ observations. Note that for $i \geq 1$, z_i (and θ_i) takes a new value with probability $\dfrac{c}{c+i}$, hence incrementing k in one. Thus

$$\mathbb{E}(k | c, n+1) = \sum_{i=1}^{n+1} \frac{c}{c+i} \cong c \log\left(\frac{n+1}{c}\right) \quad \text{as} \quad n \to \infty. \tag{3.15}$$

This indicates that the number of clusters grows logarithmically in the number of observations, hence we can expect the number of clusters k to be far smaller than the number of observations $n+1$.

3.1.5 Stick-breaking representation

The constructive representation of the Dirichlet process is often called the stick-breaking representation, Sethuraman (1994). Let $G \sim \mathrm{DP}(c, G_0)$, we can construct the discrete random measure G via

$$\begin{aligned}
&[v_1, v_2, \ldots, |c] \sim_{\text{i.i.d.}} \mathrm{Be}(\cdot|1, c), \\
&\quad \Rightarrow w_1 = v_1,\ w_j = v_j \prod_{l=1}^{j-1}(1 - v_l) \text{ for } j \geq 2, \qquad (3.16) \\
&[\theta_1, \theta_2, \ldots,] \sim_{\text{i.i.d.}} G_0, \\
&\quad \Rightarrow G(\cdot) = \sum_{j=1}^{\infty} w_j \delta_{\theta_j}(\cdot). \qquad (3.17)
\end{aligned}$$

Remark 3.9. In (3.16) notice that

$$\sum_{j=1}^{k} w_j \xrightarrow[k\to\infty]{} 1$$

with probability one. Further, note that for $j = 1, 2, \ldots$

$$\mathbb{E}(w_j) = \left(\frac{1}{c+1}\right)\left(\frac{c}{c+1}\right)^{j-1} \Rightarrow \mathbb{E}(w_1) \geq \mathbb{E}(w_2) \geq \cdots \geq \mathbb{E}(w_j) \xrightarrow[j\to\infty]{} 0,$$

so the weights are decreasing in expectation, and we say that they are weakly identifiable.

Truncating the infinite mixture (3.17), up to an integer k, such that $\sum_{j=1}^{k} w_j \approx 1$ we can sample $G \sim \mathrm{DP}(c, G_0)$ approximately. Figure 3.1 displays the probability mass function of $G \sim \mathrm{DP}(c, \mathrm{N}(0,1))$ along with the density of the base distribution. Different values of c are being plotted: $c = 0.5, 1, 5$ and 10.

Remark 3.10. Figure 3.1 stresses the importance of the concentration parameter c. When c is small, then G tends to concentrate its mass on a few atoms, see Figure 3.1

graphic a): we fixed $c = 0.5$ and to obtain $\sum_{j=1}^{k} w_j \approx 1$ we set $k = 8$. On the other hand, when c is large then G is a distribution with many atoms spread all over the support of the base distribution, see graphic d): for $c = 10$, to obtain $\sum_{j=1}^{k} w_j \approx 1$ we set $k = 180$. Finally, note that for large values of c the c. d. f. of G gets closer to the c. d. f. of the base distribution (graphic not shown).

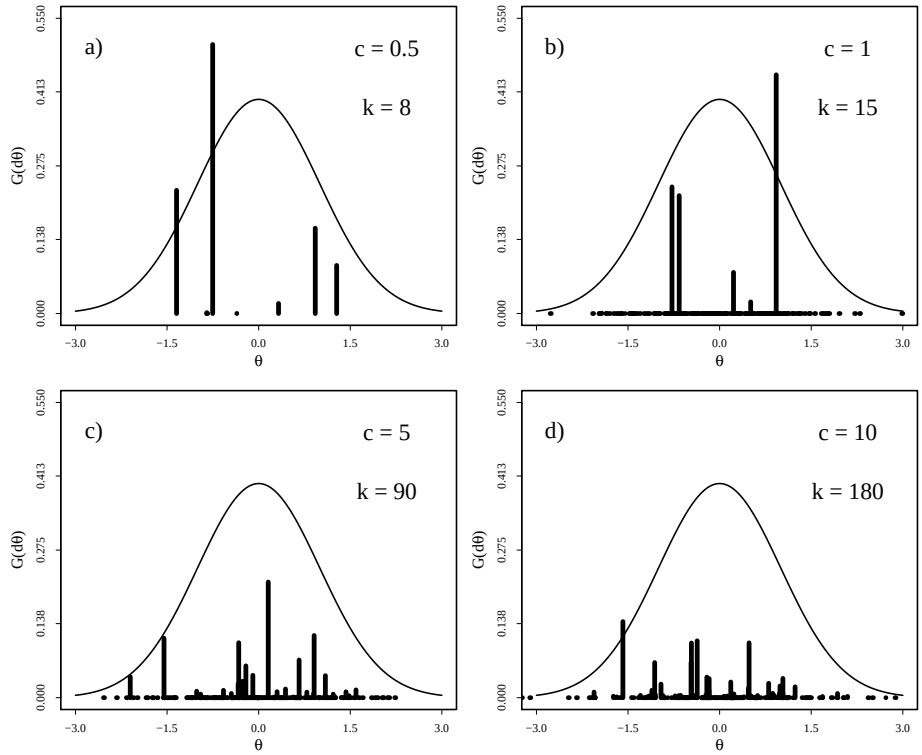

Figure 3.1: Probability mass function of $G \sim \mathrm{DP}(c, \mathrm{N}(0, 1))$ for different values of c.

For a more comprehensive summary of properties of the Dirichlet process, see Ghosal (2010), and for nonparametric models beyond the Dirichlet process see Walker, Damien, Laud, and Smith (1999) and Lijoi and Prünster (2010).

3.2 The mixture of Dirichlet process model

The mixture of Dirichlet process model (MDP), Lo (1984), consists in mixing a kernel $k(y|\theta)$ with respect to a random distribution function G, from some space of distribution

functions Ψ, to define a random density:

$$f_G(y) = \int_\Theta k(y|\theta)G(d\theta). \tag{3.18}$$

Provided that the kernel $k(y|\theta)$ is a density for each θ, then $f_G(y)$ will be a density. The original aim under the MDP model was to generate a density estimate for non-standard density functions, see Lo (1984). When G varies on Ψ, the $f_G(y)$'s define a family of random density functions, let us say $\mathscr{L}$. The key point is that $\mathscr{L}$ should be rich enough to cover the sub-space where the target density is, which is not known a-*priori*. Hence, the best way to proceed is to make $\mathscr{L}$ as large as possible, and this is achieved by taking Ψ as the space of discrete distribution functions. Thus a convenient prior for G is the Dirichlet process: $G \sim \mathrm{DP}(c, G_0(\cdot|\gamma))$ where γ are the parameters of G_0. Hence, using Sethuraman's stick-breaking representation of the Dirichlet process (3.17), we can write

$$f_G(y) = \sum_{j=1}^{\infty} w_j k(y|\theta_j), \tag{3.19}$$

and for a suitable choice of $k(y|\theta)$, $\mathscr{L}$ will cover a broad range of density functions.

Alternatively, the MDP model can be written hierarchically as follows

$$\begin{aligned} &c \sim p(c) \quad \text{and} \quad \gamma \sim p(\gamma), \\ &[G|c,\gamma] \sim \mathrm{DP}(c, G_0(\cdot|\gamma)), \\ &[\theta_1, \ldots, \theta_n|G] \sim_{\text{i.i.d.}} G, \\ &[y_i|\theta_i] \sim k(\cdot|\theta_i), \quad i = 1, \ldots, n, \end{aligned} \tag{3.20}$$

so we will model data $\mathbf{y} = \{y_i\}_{i=1}^n$ using the latent parameters $\theta_1, \ldots, \theta_n$.

It is not straightforward to sample from the posterior distribution of the MDP model, and MCMC strategies are the preferred choice. There are two main ideas to tackle this problem, and these are related to the two representations of the Dirichlet process:

- **Marginal methods:** these methods exploit Pólya's urn representation of the Dirichlet process (3.12). Thus the aim is to integrate out analytically the random

distribution G, and work with a finite model. These strategies are often described hierarchically as in (3.20), writing the random distribution G explicitly.

- **Conditional methods:** these algorithms adopt Sethuraman's stick-breaking representation of the Dirichlet process (3.17), and then work directly with the infinite mixture (3.19). It is clear that under this perspective some ideas and interpretation of the finite mixture model can be borrowed. However, we will need to deal with the infinite number of components and with the stick-breaking representation of the weights (3.16).

3.2.1 Marginal algorithms

The first algorithm of this kind is due to Escobar (1988) and then it was extended and formally published in Escobar and West (1995). Using an infinite mixture of normals, and via a Gibbs sampler Escobar and West (1995) generated density estimates, estimated the number of components of the mixture, the number of modes within the data and even made inference about the concentration parameter c of the Dirichlet process.

From this perspective, (3.20), a clustering structure can be defined by the k distinct values $\varphi_1, \ldots, \varphi_k$ of $\theta_1, \ldots, \theta_n$: G is a discrete measure and it generates ties. Thus it is clear that for different realizations of the experiment the number of distinct values may change. Section 3.1.4 is key to understand the underlying ideas for the modeling of k under this setting. Note in (3.15) the expected number of groups depends on c and n, the concentration parameter of the Dirichlet process and the number of observations respectively. For a fixed n, larger values of c will favor larger number of groups, and for a fixed c, larger number of observations will favor more groups. Also note that with (3.20) it is easy to see that additional flexibility can be included in the model by imposing priors over c and γ.

3.2.1.1 Escobar and West (1995)

We start integrating out G from the hierarchical model (3.20), via the Pólya urn representation of the Dirichlet process (3.12). Hence, it is possible to write the joint for the latent parameters directly:

$$\begin{aligned} p(\theta_1, \ldots, \theta_n|\gamma, c) &= \prod_{i=1}^{n} p(\theta_i|\theta_1, \ldots, \theta_{i-1}, \gamma, c), \\ &= \prod_{i=1}^{n} \left\{ \frac{c g_0(\theta_i|\gamma) + \sum_{j=1}^{i-1} \delta_{\theta_j}(\theta_i)}{c + i - 1} \right\}, \end{aligned} \tag{3.21}$$

where g_0 is the density function corresponding to G_0.

We do not have to worry about G anymore, and (3.20) becomes a finite model:

$$\begin{aligned} & c \sim p(c) \quad \text{and} \quad \gamma \sim p(\gamma), \\ & [\theta_1, \ldots, \theta_n|\gamma, c] \sim \prod_{i=1}^{n} \left\{ \frac{c g_0(\theta_i|\gamma) + \sum_{j=1}^{i-1} \delta_{\theta_j}(\theta_i)}{c + i - 1} \right\}, \\ & [y_i|\theta_i] \sim k(\cdot|\theta_i), \quad i = 1, \ldots, n. \end{aligned} \tag{3.22}$$

From (3.21), the full joint conditionals for the Gibbs sampler can be expressed as

$$\begin{aligned} p(\theta_i|\boldsymbol{\theta}_{-i}, \gamma, c, \mathbf{y}) &\propto k(y_i|\theta_i) \left\{ c g_0(\theta_i|\gamma) + \sum_{j \neq i} \delta_{\theta_j}(\theta_i) \right\}, \\ &\propto r_0 g(\theta_i|y_i, \gamma) + \sum_{j \neq i} r_{ij} \delta_{\theta_j}(\theta_i), \end{aligned} \tag{3.23}$$

with

$$g(\theta_i|y_i, \gamma) = \frac{k(y_i|\theta_i) g_0(\theta_i|\gamma)}{\int k(y_i|\theta_i) g_0(\theta_i|\gamma) \mathrm{d}\theta_i}, \tag{3.24}$$

and where $\mathbf{y} = \{y_i\}_{i=1}^n$ and $\boldsymbol{\theta}_{-i} = (\theta_1, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_n)$.

To sample from the mixture (3.23), we need to ensure the normalizing weights sum up to one, i.e. $r_0 + \sum_{j \neq i} r_{ij} = 1$, and this is achieved from

$$b = c \int k(y_i|\theta_i) g_0(\theta_i|\gamma) \mathrm{d}\theta_i + \sum_{j \neq i} k(y_i, \theta_j), \tag{3.25}$$

and then taking

$$\begin{aligned} r_0 &= \frac{c}{b}\int k(y_i|\theta_i)g_0(\theta_i|\gamma)\mathrm{d}\theta_i, \\ r_{ij} &= \frac{1}{b}k(y_i|\theta_j), \quad \text{for } j \neq i. \end{aligned}$$

Remark 3.11. The calculation of the normalizing constant (3.25) is feasible when $g_0(\theta|\gamma)$ is a conjugate prior for $k(y|\theta)$, and this was the setting of Escobar and West (1995): they use a normal kernel and assigned a normal/inverse-gamma distribution to $g_0(\theta|\gamma)$. For non-conjugate pairs this scheme becomes rather complicated.

To make inference about the concentration parameter c, of the Dirichlet process, Escobar and West borrowed the implied prior for k, $p(k|c,n)$, from Antoniak (1974), and updated c via $p(c|k,n) \propto p(c)p(k|c,n)$. A concise description of this algorithm is given in Algorithm 3.2.

Algorithm 3.2 Escobar and West (1995): marginal strategy

Require: Given $c^t, k^t, (\theta_1^t, \ldots, \theta_n^t)$, simulate $c^{t+1}, k^{t+1}, (\theta_1^{t+1}, \ldots, \theta_n^{t+1})$ via

1: **for** $i = 1$ to n **do**

2:

$$\theta_i^{t+1} = \begin{cases} \theta_j^t & \text{with probability } \ r_{ij} \ (j \neq i), \\ \theta_i \sim g(\theta_i|y_i,\gamma) & \text{with probability } \ r_0. \end{cases}$$

3: **end for**

4: Update k^{t+1}: k^{t+1} is the number of distinct values among $\theta_1^{t+1}, \ldots, \theta_n^{t+1}$.

5: $c^{t+1} \sim p(c|k^{t+1},n) \propto p(c)p(k^{t+1}|c,n) \propto p(c)\left\{c^{k^{t+1}}\dfrac{\Gamma(c)}{\Gamma(c+n)}\right\}$.

The number of components of the mixture k is recorded at each iteration of the algorithm, and to make inference about it we proceed as in the finite mixture model (2.37). To generate a predictive density estimate we approximate

$$p(y^*|\mathbf{y}) \approx \frac{1}{N}\sum_{t=1}^{N} k(y^*|\theta^*),$$

for $t = 1, \ldots, N$ (N is the number of iterations after a burn in period), where θ^* is drawn from (3.12).

Escobar and West (1995) also demonstrated that the Markov chain generated via their strategy converges to the correct posterior distribution, and that it is irreducible and aperiodic. From Lemma A.3.1 in Appendix A the chain is ergodic. However, despite this result, the algorithm shows poor mixing, and thus convergence to the posterior distribution is rather slow. We have borrowed a quote from Neal (2000) that describes this in detail:

> The problem is that there are often groups of observations that with high probability are associated with the same θ. Since the algorithm cannot change the θ for more than one observation simultaneously, a change to the θ values for observations in such a group can occur only rarely, as such a change requires passage through a low-probability intermediate state in which observations in the group do not all have the same θ value.

3.2.1.2 MacEachern (1994)

To improve the convergence problems of Algorithm 3.2, MacEachern (1994) developed a more efficient approach. This time the idea is to use the clustering properties of the Dirichlet process, and update the Markov chain via the discrete latent variables $\mathbf{z}$, see Section 3.1.4.

First, G is integrated out from (3.20), but now via (3.13), in this case

$$p(\theta_1, \ldots, \theta_n | \gamma, c) = \prod_{i=1}^{n} \left\{ \frac{c g_0(\theta_i|\gamma) + \sum_{j=1}^{k} n_j \delta_{\varphi_j}(\theta_i)}{c + i - 1} \right\}, \tag{3.26}$$

Hence, the full joint conditionals for the Gibbs sampler can be expressed as

$$\begin{aligned} p(\theta_i | \boldsymbol{\theta}_{-i}, \mathbf{z}_{-i}, \gamma, c, y_i) &\propto k(y_i|\theta_i) \left\{ c g_0(\theta_i|\gamma) + \sum_{j \in A_{-i}} n_{-i,j} \delta_{\varphi_j}(\theta_i) \right\}, \\ &\propto r_0 g(\theta_i|\gamma, y_i) + \sum_{j \in A_{-i}} r_{ij} \delta_{\varphi_j}(\theta_i), \end{aligned} \tag{3.27}$$

where $\boldsymbol{\theta}_{-i}$ is as before, $\mathbf{z}_{-i} = (z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n)$, A_{-i} = set of unique values of $\mathbf{z}_{-i}$, $n_{-i,j} = \sum_{l \neq i} \delta_j(z_l)$, and $g(\theta_i|\gamma, y_i)$ is as in (3.24)

Again in (3.27) we need to ensure $r_0 + \sum_{j \in A_{-i}} r_{ij} = 1$, and this is achieved by calculating

$$b = c \int k(y_i|\theta) g_0(\theta|\gamma) \mathrm{d}\theta + \sum_{j \in A_{-i}} n_{-i,j} k(y_i|\varphi_j),$$

and then

$$r_0 = \frac{c}{b} \int k(y_i|\theta) g_0(\theta|\gamma) \mathrm{d}\theta, \tag{3.28}$$

$$r_{ij} = \frac{1}{b} n_{-i,j} k(y_i|\varphi_j) \text{ for } j \in A_{-i}. \tag{3.29}$$

Remark 3.12. Observe that (3.27) defines the discrete variable z_i, where

$$p(z_i|\boldsymbol{\theta}_{-i}, \mathbf{z}_{-i}, \gamma, c, y_i) = r_0 \delta_{k+1}(z_i) + \sum_{j \in A_{-i}} r_{ij} \delta_j(z_i)$$

Hence, the probability of to open a new group is given by (3.28), and to allocate an observation to one of the existing groups by (3.29). Further, note that if z_i is sampled first, then θ_i can be easily updated.

Remark 3.13. To update the variables $\varphi_1, \ldots, \varphi_k$ we use the latent allocations and proceed exactly as in the finite mixture model, see Algorithm 2.1.

A concise description of MacEachern's algorithm is given in Algorithm 3.3. The problem with this strategy lies in the fact that if $g_0(\theta|\gamma)$ is not chosen as a conjugate prior for $k(y|\theta)$, then it will be costly to integrate out θ_i numerically.

3.2.1.3 Neal (2000)

It is clear how the last two algorithms integrate out the random distribution, G, from the MDP model via the Pólya urn representation of the Dirichlet process (3.12). However, Neal (2000) developed an alternative view that is useful for deriving algorithms to sample from the posterior distribution for the MDP model. In this case we start with a

Algorithm 3.3 MacEachern (1994): marginal strategy

Require: Given $c^t, \gamma^t, k^t, (\theta_1^t, \ldots, \theta_n^t), (\varphi_1^t, \ldots, \varphi_{k^t}^t), \mathbf{z}^t$, simulate c^{t+1}, γ^{t+1}, k^{t+1}, $(\theta_1^{t+1}, \ldots, \theta_n^{t+1})$, $(\varphi_1^{t+1}, \ldots, \varphi_{k^{t+1}}^{t+1})$, $\mathbf{z}^{t+1}$ via

1: $k^{t+1} = k^t$

2: **for** $i = 1$ to n **do**

3: Find A_{-i}: A_{-i} = set of unique values in $(z_1^{t+1}, z_2^{t+1}, \ldots, z_{i-1}^{t+1}, z_{i+1}^t, \ldots, z_n^t)$.

4: $(j \in A_{-i} \bigcup \{k^{t+1} + 1\})$

$$z_i^{t+1} \sim p(z_i = j | \cdots) = \begin{cases} r_{ij} & \text{if } j \in A_{-i}, \\ r_0 & \text{otherwise.} \end{cases}$$

$$\theta_i^{t+1} = \begin{cases} \varphi_{z_i^{t+1}} & \text{if } z_i^{t+1} \in A_{-i}, \\ \theta_i \sim g(\theta_i | y_i, \gamma^t) & \text{otherwise.} \end{cases}$$

5: Update k^{t+1}: k^{t+1} = #set of unique values in $A_{-i} \bigcup \{z_i^{t+1}\}$.

6: **end for**

7: **for** $j = 1, \ldots, k^{t+1}$ **do**

8: $\varphi_j^{t+1} \sim p(\varphi_j | \mathbf{z}^{t+1}, \gamma^t, \mathbf{y}) \propto g_0(\varphi_j | \gamma^t) \left\{ \prod_{\{i: z_i^{t+1} = j\}} k(y_i | \varphi_j) \right\}$,

9: **end for**

10: $\gamma^{t+1} \sim p(\gamma | \varphi_1^{t+1}, \ldots, \varphi_{k^{t+1}}^{t+1}, k^{t+1}) \propto p(\gamma) \left\{ \prod_{j=1}^{k^{t+1}} g_0(\varphi_j^{t+1} | \gamma) \right\}$,

11: $c^{t+1} \sim p(c | k^{t+1}, n) \propto p(c) p(k^{t+1} | c, n) \propto p(c) \left\{ c^{k^{t+1}} \dfrac{\Gamma(c)}{\Gamma(c+n)} \right\}$.

hierarchical representation of the finite mixture model:

$$\begin{aligned}
&c \sim p(c) \quad \text{and} \quad \gamma \sim p(\gamma), \\
&[\mathbf{w}|\delta, k] \sim \text{Dir}\left(\mathbf{w}\middle|\frac{c}{k}, \ldots, \frac{c}{k}\right), \\
&[\varphi_j|\gamma, k] \sim g_0(\cdot|\gamma), \ j = 1, 2, \ldots, k, \\
&[z_i|\mathbf{w}, k] \sim \sum_{j=1}^{k} w_j \delta_j(\cdot), \quad i = 1, \ldots, n, \\
&[y_i|\boldsymbol{\varphi}, \mathbf{z}] \sim k(\cdot|\varphi_{z_i}), \quad i = 1, \ldots, n.
\end{aligned} \tag{3.30}$$

Remark 3.14. Note that (3.30) is a particular case of the finite mixture model (2.5), and in this case the prior over the finite weights is a symmetric Dirichlet distribution with parameter c/k. Hence, we can use the MCMC techniques of Chapter 2 to generate a sample from the correct posterior distribution of this model.

Remark 3.15. In this case we will work directly with the set of distinct values $\boldsymbol{\varphi} = (\varphi_1, \ldots, \varphi_k)$ of $(\theta_1, \ldots, \theta_n)$ $(k \leq n)$.

It is possible to integrate out the finite weights from (3.30) and derive the prior for the z_i, and this is of the form (see p. 251 of Neal (2000)):

$$p(z_i = j|c, i-1, z_1, \ldots, z_{i-1}) = \frac{n_{i,j} + c/k}{c + i - 1},$$

where $n_{i,j} = \sum_{l=1}^{i} \delta_j(z_l)$.

If we now let k go to infinity, the conditional probabilities reach the following limits

$$p(z_i = j|c, i-1, z_1, \ldots, z_{i-1}) = \begin{cases} \dfrac{c}{c + i - 1}, & \text{if } j \notin \{z_1, \ldots, z_i\}, \\ \dfrac{n_{i,j}}{c + i - 1}, & \text{for } j \in \{z_1, \ldots, z_i\}. \end{cases} \tag{3.31}$$

But these conditional probabilities are equivalent to those obtained following Pólya's urn representation of the Dirichlet process, see expressions (3.13) and (3.14). Hence, from the remark 3.8 we know that the conditional probabilities (3.31) are all that is

needed to define the DP model and thus the MDP model via (3.30), and this is the main argument of Neal (2000).

To devise a Gibbs sampler following this alternative view, we need to derive the conditional probabilities to update the allocations. Thus, going back to the finite mixture model (3.30), it is not difficult to see that these are given by

$$p(z_i = j|\boldsymbol{\varphi}, \mathbf{z}_{-i}, c, n, y_i) = bk(y_i|\varphi_j)\frac{n_{-i,j} + c/k}{c + n - 1},$$

where $n_{-i,j}$ and $\mathbf{z}_{-i}$ are as in the past algorithms, and b is the normalizing constant.

Using the same reasoning as before, we let k go to infinity to obtain

$$p(z_i = j|\boldsymbol{\varphi}, \mathbf{z}_{-i}, \gamma, c, n, y_i) = \begin{cases} b\int k(y_i|\varphi)g_0(\varphi|\gamma)\mathrm{d}\varphi\dfrac{c}{c+n-1}, & \text{if } j \notin \mathbf{z}_{-i}, \\ bk(y_i|\varphi_j)\dfrac{n_{-i,j}}{c+n-1}, & \text{for } j \in \mathbf{z}_{-i}. \end{cases} \tag{3.32}$$

This is Algorithm 2 of Neal (2000), and a concise description is given in Algorithm 3.4 ($g(\theta|y_i, \gamma)$ is as in the previous strategies). However, this method again relies on $g_0(\theta|\gamma)$ being chosen as a conjugate prior for $k(y|\theta)$. An alternative to solve this problem is provided by Algorithm 8 of Neal (2000), and here ideas similar to those of slice sampling methods are used (see Section A.4.3 in the Appendix).

The basic idea is as follows. Imagine that a sample from $p(x)$ is needed, hence we introduce the auxiliary variable y and sample from $p(x, y)$ instead. We will need to sample from the conditional distributions $p(y|x)$ and $p(x|y)$, then we keep only the variables of interest (the auxiliary variables are discarded). This strategy will clearly produce a sample from $p(x)$.

This idea can be used to update the z_i for an MDP model without having to integrate with respect $g_0(\theta|\gamma)$. The permanent state of the Markov chain will consist of the z_i and the $\boldsymbol{\varphi}$, as in the previous algorithm, but when z_i is updated temporary auxiliary variables are introduced that represent possible values for the parameters of components that are not associated with any other observations. Thus z_i is updated via a Gibbs sampler

with respect to the distribution that includes these auxiliary variables. For a complete description see Algorithm 8 of Neal (2000).

Algorithm 3.4 Neal (2000)

Require: Given $c^t, \gamma^t, k^t, (\theta_1^t, \ldots, \theta_n^t), (\varphi_1^t, \ldots, \varphi_{k^t}^t), \mathbf{z}^t$, simulate c^{t+1}, γ^{t+1}, k^{t+1}, $(\theta_1^{t+1}, \ldots, \theta_n^{t+1})$, $(\varphi_1^{t+1}, \ldots, \varphi_{k^{t+1}}^{t+1})$, $\mathbf{z}^{t+1}$ via

1: $k^{t+1} = k^t$

2: **for** $i = 1$ to n **do**

3: Find A_{-i}: A_{-i} = set of unique values in $(z_1^{t+1}, z_2^{t+1}, \ldots, z_{i-1}^{t+1}, z_{i+1}^t, \ldots, z_n^t)$.

4: $(j \in A_{-i} \bigcup \{k^{t+1} + 1\})$

Sample z_i^{t+1} as defined by equation (3.32).

$$\theta_i^{t+1} = \begin{cases} \varphi_{z_i^{t+1}} & \text{if } z_i^{t+1} \in A_{-i}, \\ \theta_i \sim g(\theta | y_i, \gamma^t) & \text{otherwise.} \end{cases}$$

5: Update k^{t+1}: k^{t+1} = #set of unique values in $A_{-i} \bigcup \{z_i^{t+1}\}$.

6: **end for**

7: **for** $j = 1, \ldots, k^{t+1}$ **do**

8: $\varphi_j^{t+1} \sim p(\varphi_j | \mathbf{z}^{t+1}, \gamma^t, \mathbf{y}) \propto g_0(\varphi_j | \gamma^t) \left\{ \prod_{\{i: z_i^{t+1} = j\}} k(y_i | \varphi_j) \right\}$,

9: **end for**

10: $\gamma^{t+1} \sim p(\gamma | \varphi_1^{t+1}, \ldots, \varphi_{k^{t+1}}^{t+1}, k^{t+1}) \propto p(\gamma) \left\{ \prod_{j=1}^{k^{t+1}} g_0(\varphi_j^{t+1} | \gamma) \right\}$,

11: $c^{t+1} \sim p(c | k^{t+1}, n) \propto p(c) p(k^{t+1} | c, n) \propto p(c) \left\{ c^{k^{t+1}} \dfrac{\Gamma(c)}{\Gamma(c+n)} \right\}$.

3.2.1.4 Convergence

On the convergence side, the strategy of MacEachern (1994) and Neal (2000) generate ergodic Markov chains, and they exhibit better mixing than that designed by Escobar and West (1995). However, in both cases there are concerns about the mixing over

the posterior distribution of the allocations $\mathbf{z}$, which can be poor due to the single-site updating scheme $z_i|\mathbf{z}_{-i}$. This has been discussed for example by Griffin and Holmes (2010), we follow their ideas to explain this point in detail. Imagine that for some $\mathbf{z}$ the clusters defined by $A_j = \{i : z_i = j\}$, for $j = 1, \ldots, 4$, are well supported by the data. A single-site updating scheme can only move between $A_1, \ldots, A_4$ by visiting "intermediate clusters" that differ by moving a single observation from one group to the other. This scheme will mix poorly if there are no well-supported intermediate clusters, since only one observation can be moved between groups at a time.

Remark 3.16. The poor mixing described for single-site updating schemes is similar to that of the Gibbs sampler for finite mixtures, see Section 2.2.5.

This problem can be addressed using split and combine moves, here a cluster is split or two clusters are merged to propose a new configuration of clusters. This proposal is either accepted or rejected using the Metropolis-Hastings acceptance probability. The problem now is how to generate proposals with reasonable chance of being accepted. See Griffin and Holmes (2010) for details about these alternative strategies.

3.2.2 Conditional algorithms

The Pólya urn representation of the Dirichlet process provides mechanisms to design MCMC samplers to make inference about the posterior distribution of the MDP model. However, Lo's model, in conjunction with the Dirichlet process representation of Sethuraman (1994), offers the possibility to explore alternative ideas. This has led to interest in methods which avoid integrating out the random measure G and work directly with the infinite mixture (3.19). Under this setting, latent allocation variables can be included: $\mathbf{z} = \{z_i\}_{i=1}^n$, such that given z_i, the component from which y_i has been drawn is known, i.e. $p(y_i|\boldsymbol{\theta}, z_i) = k(y_i|\theta_{z_i})$. With the introduction of these latent variables, the

infinite mixture model can be written hierarchically as follows

$$
\begin{aligned}
& c \sim p(\cdot|\lambda) \ \text{ and } \ \gamma \sim p(\cdot|\zeta), \\
& [v_1, v_2 \ldots, |c] \sim_{\text{i.i.d.}} \text{Be}(\cdot|1, c), \\
& \qquad \Rightarrow w_1 = v_1 \text{ and } w_j = v_j \prod_{l=1}^{j-1}(1 - v_l) \text{ for } j \geq 2, \\
& [\theta_1, \theta_2, \ldots, |\gamma] \sim_{\text{i.i.d.}} G_0(\cdot|\gamma), \\
& [z_1, \ldots, z_n|\mathbf{w}] \sim_{\text{i.i.d.}} \sum_{j=1}^{\infty} w_j \delta_j(\cdot), \\
& [y_1, \ldots, y_n|\theta, \mathbf{z}] \sim_{\text{i.i.d.}} k(\cdot|\theta_{z_i}).
\end{aligned} \tag{3.33}
$$

where $\theta = \{\theta_j\}_{j=1}^{\infty}$ and $\mathbf{w} = \{w_j\}_{j=1}^{\infty}$.

Since (3.19) defines an infinite mixture model and because the w_j's decrease exponentially quickly in expectation (see remark 3.9) only a small number of components, say k, will be used to model the data. This is the key difference between the infinite model (3.19) and the finite mixture model (2.1). In the finite mixture model, the number of components to model the data is fixed and we need to devise the case for an unknown number of components separately. For the infinite model k changes naturally, then, to make inference for the number of groups within the data we can take advantage of this characteristic to estimate $p(k|\mathbf{y})$. So we can use (3.19) for the modeling of clusters or to generate density estimates of non-standard distributions.

Under this setting, (3.19) or (3.33), it is important to remark that the concentration parameter c, of the Dirichlet process, plays the same role as in the Pólya urn representation. To note this from this perspective, see Figure 3.1 and remark 3.10. From the remark 3.10 it is clear that large values of c will favor larger values of k. The converse is true, small values of c will support smaller values of k. This is for the modeling of clusters, but it has the exact same effect in a density estimation context. Under this framework the concentration parameter c is sometimes called the smoothing parameter: for larger values of c smoother density estimates are obtained.

There are three possible approaches to deal with the infinite mixture: truncation, retrospective sampler and slice sampler. It is important to stress that all the methods change the infinite mixture (3.19) for a finite mixture; the difference between the three methods is how this is achieved. Thus leading to a finite-dimensional posterior distribution which can be simulated using similar methods to those described for the finite mixture model. Assuming a finite mixture with k components, it is easy to write the corresponding MCMC strategy, this is shown in Algorithm 3.5.

Remark 3.17. The key difference between the three proposals, to change the infinite mixture for a finite model, lie in steps 7 and 9 of Algorithm 3.5. Note that, without the assumption of a finite model, there should be an infinite choice for z_i, i.e.

$$p(z_i = j|\mathbf{w}, \boldsymbol{\theta}, \mathbf{y}) = \frac{w_j k(y_i|\theta_j)}{\sum_{j=1}^{\infty} w_j k(y_i|\theta_j)} \quad (j = 1, 2, \ldots,). \tag{3.34}$$

It is straightforward to justify all the steps in Algorithm 3.5. First note that Steps 2 and 8 follow directly from the finite model (2.1), and the Gibbs sampler described in Algorithm 2.1. Step 11 is derived from the ideas of Escobar and West (1995), to update the concentration parameter of the Dirichlet process. Finally, the full joint conditional for v_j, in Step 3, is obtained via

$$p(v_j|\mathbf{z}, c) \propto \text{Be}(v_j|1, c) \prod_{i=1}^{n} w_{z_i} \propto (1 - v_j)^{c-1} \prod_{l=1}^{k} w_l^{n_l},$$

where from (3.16)

$$w_l^{n_l} = \begin{cases} v_1^{n_1} & \text{if } l = 1, \\ v_l^{n_l} \left\{ \prod_{h=1}^{l-1} (1 - v_h)^{n_l} \right\} & \text{for } l \geq 2. \end{cases}$$

Thus (note that $n = \sum_{l=1}^{k} n_l$),

$$\begin{aligned} p(v_j|\mathbf{z}, c) &\propto (1-v_j)^{c-1}\left[v_1^{n_1}\prod_{l=2}^{k} v_l^{n_l}\left\{\prod_{h=1}^{l-1}(1-v_h)^{n_l}\right\}\right], \\ &\propto (1-v_j)^{c-1}\left\{v_j^{n_j}(1-v_j)^{n_{j+1}}\cdots(1-v_j)^{n_k}\right\}, \\ &\propto v_j^{n_j+1-1}(1-v_j)^{n-n_1+\cdots+n_j+c-1}, \\ &\propto \mathrm{Be}\left(v_j|n_j+1, n-\sum_{l=1}^{j} n_l + c\right). \end{aligned}$$

Algorithm 3.5 Conditional method

Require: Given k^t, $\mathbf{v}^t$, $\mathbf{w}^t$,$\boldsymbol{\theta}^t$, $\mathbf{z}^t$, γ^t and c^t simulate k^{t+1}, $\mathbf{v}^{t+1}$, $\mathbf{w}^{t+1}$, $\boldsymbol{\theta}^{t+1}$, $\mathbf{z}^{t+1}$, γ^{t+1} and c^{t+1} via

1: **for** $j = 1$ to k^t **do**

2: $\theta_j^{t+1} \sim p(\theta_j|\mathbf{z}^t, \mathbf{y}) \propto p(\theta_j|\gamma^t) \prod_{\{i:z_i^t=j\}} k(y_i|\theta_j)$.

3: $v_j^{t+1} \sim p(v_j|\mathbf{z}, c^t) = \mathrm{Be}\left(n_j+1, n-\sum_{l=1}^{j} n_l + c^t\right)$.

4: Calculate $\mathbf{w}^{t+1}$ via the stick-breaking construction (3.16), i.e.

$$w_j^{t+1} = \begin{cases} v_1^{t+1} & \text{if } j = 1, \\ w_{j-1}^{t+1}\left(1-v_{j-1}^{t+1}\right)\left(v_j^{t+1}/v_{j-1}^{t+1}\right) & \text{if } j \geq 2. \end{cases}$$

5: **end for**

6: $\gamma^{t+1} \sim p(\gamma|\theta_1^{t+1}, \ldots, \theta_{k^t}^{t+1}, k^t) \propto p(\gamma)\left\{\prod_{j=1}^{k^t} g_0(\theta_j^{t+1}|\gamma)\right\}$,

7: **for** $i = 1$ to n **do**

8: $z_i^{t+1} \sim p(z_i = j|\mathbf{w}^{t+1}, \boldsymbol{\theta}^{t+1}, k^{t+1}, \mathbf{y}) \propto w_j^{t+1} k(y_i|\theta_j^{t+1}) \quad (j = 1, 2, \ldots, k^{t+1})$.

9: **end for**

10: Update k^{t+1}.

11: $c^{t+1} \sim p(c|k^{t+1}, n) \propto p(c)p(k^{t+1}|c, n) \propto p(c)\left\{c^{k^{t+1}}\dfrac{\Gamma(c)}{\Gamma(c+n)}\right\}$.

3.2.2.1 Truncation method

The truncation method replaces the infinite-dimensional random distribution function (3.17) with a finite version

$$G_k(\cdot) = \sum_{j=1}^{k} w_j \delta_{\theta_j}(\cdot), \text{ where } \sum_{j=1}^{k} w_j \approx 1. \tag{3.35}$$

The use of this proposal presents an important problem, how to choose k to obtain a good approximation of G via G_k. Truncated versions of the Dirichlet process have been considered for example by Ishwaran and James (2001). Their idea consists of selecting a value k that controls the difference between the probability of the observations under the truncated and infinite dimensional priors. The probability under G_k is given by

$$\pi_k(\mathbf{y}) = \int \left(\prod_{i=1}^{n} k(y_i|\theta_i) G_k(\mathbf{d}\theta_i) \right) \Pi_k(G_k),$$

where Π_k represents the probability law of G_k (this definition extends to the infinite-dimensional case by setting $k = \infty$). Then if $||\cdot||_1$ denotes the L_1 distance, Ishwaran and James (2001) showed that

$$||\pi_k(\mathbf{y}) - \pi_\infty(\mathbf{y})||_1 \leq 4 \left[1 - \mathbb{E} \left\{ \left(\sum_{j=1}^{k-1} w_j \right)^n \right\} \right] \approx 4n \exp\{-(k-1)/c\}. \tag{3.36}$$

Thus, from the right hand side of (3.36), k can be chosen to make the bound small. This allows the simple calculation of a value of k that gives a particular level of error.

Since a truncation is introduced, the normalizing constant in (3.34) is approximated, and in Algorithm 3.5, Step 9, the truncated method delivers a fixed $k^t = k$ for all $t = 1, \ldots, N$, and in Step 7, the same step as in the Gibbs sampler for finite mixtures is recovered.

Remark 3.18. The value of k obtained from the truncation method is potentially large. In the next sections two methods to make inference about suitable stochastic values of k are described, and these strategies provide values that are much less than the value provided by the truncation method.

3.2.2.2 Retrospective method

The retrospective method, Papaspiliopoulos and Roberts (2008), avoids the truncation by making clever use of properties of the weights derived from the stick-breaking construction (see remark 3.9), and of the inverse cumulative function (ICF) technique for simulating discrete random variables. That is, let $z \in \{1, 2, \ldots, k\}$ be a random variable such that $p(z = j) = w_j$ for $j = 1, \ldots, k$. With $0 \leq w_j \leq 1$ and $\sum_{j=1}^{k} w_j = 1$, thus we simulate $u \sim \mathrm{U}(0, 1)$ and set $z = j$ iff

$$\sum_{l=0}^{j-1} w_l < u \leq \sum_{l=1}^{j} w_l, \text{ with } w_0 = 0, \tag{3.37}$$

A formal description of the ICF technique for simulating discrete random variables is provided in Algorithm 3.6, see Ripley (1987).

Algorithm 3.6 ICF technique for simulating discrete random variables

1: $u \sim \mathrm{U}(0, 1)$. Let $j = 1$.
2: **while** $\sum_{l=1}^{j} w_l \leq u$ **do**
3: $\quad j = j + 1$.
4: **end while**
5: **return** j.

To understand how the retrospective sampler works imagine that $z \in \{1, 2, \ldots, \}$ (this is the case for (3.34)), and that the first weights accumulate more mass than the subsequent weights, as the weights generated by the stick-breaking construction (3.16). Concisely, let $G \sim \mathrm{DP}(c, G_0)$ and following retrospective sampler ideas let us generate a random sample of size n from G. This Algorithm is displayed in Algorithm 3.7.

First, the allocations $z_1, \ldots, z_n$ are sampled following the ICF technique for simulating discrete random variables. At the end of the algorithm we recover k, the maximum number of groups needed to generate n allocations, along with $w_1, \ldots, w_k$ and $\theta_1, \ldots, \theta_k$. Thus θ_{z_i}, for $i = 1, \ldots, n$, are random variables i. i. d. from G, and all the information to generate the graphic of the probability mass function of G is available.

Remark 3.19. There is no need to know a *priori* the complete set of probabilities $w_1, w_2 \ldots$, to generate $z_i = j$. It is possible to start only with w_1 and generate $w_2, w_3, \ldots, w_k$ as required. The total number of groups, k, should be updated if and only if a new group is opened, this can be seen in Algorithm 3.7, Steps 6-10. Given the structure of the weights, k will be updated only a few times during the n iterations of the algorithm. From (3.15), we can expect the number of groups k to be far smaller than the number of observations n.

Algorithm 3.7 Retrospective sampling for the Dirichlet process

1: Simulate $v_1 \sim \text{Be}(1, c)$, $w_1 = v_1$ and $\theta_1 \sim G_0$. Let $k = 1$. {Initialize k}
2: **for** $i = 1$ to n **do**
3: $u \sim \text{U}(0, 1)$. Let $j = 1$.
4: **while** $\sum_{l=1}^{j} w_l \leq u$ **do**
5: $j = j + 1$.
6: **if** $k < j$ **then**
7: $k = k + 1$. {Update the number of groups k (only if needed)}
8: $v_j \sim \text{Be}(1, c)$ and $\theta_j \sim G_0$. {Generate the extra random variables}
9: $w_j = v_j \left(\frac{1 - v_{j-1}}{v_{j-1}} \right) w_{j-1}$. {Generate the weights via (3.16)}
10: **end if**
11: **end while**
12: $z_i = j$. {Storing the sampled allocation}
13: **end for**
14: **return** $z_1, \ldots, z_n$. {Sampled allocations}
15: **return** k, $(w_1, \ldots, w_k)$, $(\theta_1, \ldots, \theta_k)$. {Information to recover G}

Via retrospective sampler ideas a sample from G is easily generated, however simulation from z_i in Algorithm 3.5 is more complicated.

Remark 3.20. It is not possible to calculate the normalizing constant for the weights of the target density, see (3.34).

To overcome this problem the obvious choice is a Metropolis-Hastings sampler, see Appendix A, Section A.4.2. The key fact is that the target density should be known up to a proportionality constant. In this framework, first, an instrumental distribution needs to be defined, second, proposals are generated using the instrumental distribution and finally these proposals are evaluated in the Metropolis-Hastings acceptance probability.

With these ideas, to update $\mathbf{z}$, we can use a composition of n Metropolis-Hastings kernels, which update each of the z_i's in turn. The strategy outlined by Papaspiliopoulos and Roberts (2008) is as follows. Let $k = \max\{\mathbf{z}\} = \max_{i=1,\ldots,n}\{z_i\}$ and

$$\mathbf{z}(i,j) = (z_1, \ldots, z_{i-1}, j, z_{i+1}, \ldots, z_n)$$

be the vector of allocations produced from $\mathbf{z}$ by substituting the i^{th} element by j. Thus, the update for each z_i is generated by proposing to move $\mathbf{z}$ to $\mathbf{z}(i,j)$, where the proposed $\mathbf{z}(i,j)$ is generated from the probability mass function

$$q(\mathbf{z}(i,j)|\mathbf{z}) = \begin{cases} w_j k(y_i|\theta_j)/b_i(\mathbf{z}), & \text{for } j \leq k, \\ w_j M_i(\mathbf{z})/b_i(\mathbf{z}), & \text{for } j > k, \end{cases}$$

where $M_i(\mathbf{z}) = \max_{j \leq k}\{k(y_i|\theta_j)\}$, and the normalizing constant is given by

$$b_i(\mathbf{z}) = \sum_{j=1}^{k} w_j k(y_i|\theta_j) + M_i(\mathbf{z}) \left(1 - \sum_{j=1}^{k} w_j\right).$$

The analogous definitions are coherent for $\mathbf{z}(i,j)$. In this case, $k^* = \max\{\mathbf{z}(i,j)\}$ will be the maximum element of $\mathbf{z}(i,j)$. Also note the variables for the proposal $\mathbf{z}(i,j)$ will be denoted by $\mathbf{v}^* = \{v_j^*\}_{j=1}^{k^*}$ $\mathbf{w}^* = \{w_j^*\}_{j=1}^{k^*}$ and $\theta^* = \{\theta_j^*\}_{j=1}^{k^*}$.

The variables for the proposals are generated via

$$\begin{aligned} \theta_j^* &\sim p(\theta_j^*|\gamma) \prod_{\{i:z_i=j\}} k(y_i|\theta_j^*) \text{ if } j \leq k \text{ or } \sim p(\theta_j^*|\gamma) \text{ otherwise,} \\ v_j^* &\sim \text{Be}\left(n_j + 1, n - \sum_{l=1}^{j} n_l + c\right) \text{ if } j \leq k \text{ or } \sim \text{Be}(1,c) \text{ otherwise,} \\ &\Rightarrow w_j^* = w_{j-1}^* \left(1 - v_{j-1}^*\right)\left(v_j^*/v_{j-1}^*\right). \end{aligned}$$

Remark 3.21. If $j > k$ the additional random variables are sampled from the priors, only if needed, and this is why retrospective sampler ideas work under this framework.

Defining

$$\pi_{\mathbf{z},\mathbf{z}(i,j)} = \frac{p(z_i = j|\mathbf{w}^*, \theta^*, \mathbf{y})q(\mathbf{z}|\mathbf{z}(i,j))}{p(z_i|\mathbf{w}, \theta, \mathbf{y})q(\mathbf{z}(i,j)|\mathbf{z})},$$

the acceptance probability for the Metropolis-Hastings can be derived considering three cases:

$$\pi_{\mathbf{z},\mathbf{z}(i,j)} = \begin{cases} \left(\frac{w_j^* k(y_i|\theta_j^*)}{w_{z_i} k(y_i|\theta_{z_i})}\right)\left(\frac{w_{z_i} k(y_i|\theta_{z_i}) b_i(\mathbf{z})}{w_j^* k(y_i|\theta_j^*) b_i(\mathbf{z}(i,j))}\right), & \text{if } j \leq k \text{ and } k = k^*, \\ \left(\frac{w_j^* k(y_i|\theta_j^*)}{w_{z_i} k(y_i|\theta_{z_i})}\right)\left(\frac{w_{z_i} M_i(\mathbf{z}(i,j)) b_i(\mathbf{z})}{w_j^* k(y_i|\theta_j^*) b_i(\mathbf{z}(i,j))}\right), & \text{if } j \leq k \text{ and } k^* < k, \\ \left(\frac{w_j^* k(y_i|\theta_j^*)}{w_{z_i} k(y_i|\theta_{z_i})}\right)\left(\frac{w_{z_i} k(y_i|\theta_{z_i}) b_i(\mathbf{z})}{w_j^* M_i(\mathbf{z}) b_i(\mathbf{z}(i,j))}\right), & \text{if } j \leq k^* \text{ and } k < k^*. \end{cases}$$

Canceling the respective terms in each case, the acceptance probability is given by

$$\alpha(\mathbf{z}, \mathbf{z}(i,j)) = \begin{cases} 1, & \text{if } j \leq k \text{ and } k = k^*, \\ \min\left\{1, \frac{M_i(\mathbf{z}(i,j)) b_i(\mathbf{z})}{k(y_i|\theta_{z_i}) b_i(\mathbf{z}(i,j))}\right\}, & \text{if } j \leq k \text{ and } k^* < k, \\ \min\left\{1, \frac{k(y_i|\theta_j^*) b_i(\mathbf{z})}{M_i(\mathbf{z}) b_i(\mathbf{z}(i,j))}\right\}, & \text{if } j \leq k^* \text{ and } k < k^*. \end{cases} \tag{3.38}$$

The retrospective sampler strategy to solve Steps 7-10 of Algorithm 3.5, is summarized in Algorithm 3.8.

3.2.2.3 Slice sampler

A second method to sample from (3.34) without truncation error was first described by Walker (2007), and later improved and extended by Kalli, Griffin, and Walker (2011). This method is based on slice sample schemes (Damien, Wakefield, and Walker (1999)), see Section A.4.3. The idea is to add for every z_i a slice variable u_i such that

$$p(z_i, u_i|\mathbf{w}, \theta, \mathbf{y}) \propto \mathrm{U}(u_i|0, \xi_{z_i}) w_{z_i} k(y_i|\theta_{z_i}). \tag{3.39}$$

For every positive sequence $\xi_1, \xi_2, \ldots$ expression (3.39) is a valid joint distribution, also note that integrating u_i we go back to the original distribution (3.34).

Algorithm 3.8 Retrospective sampler

Require: k, $\mathbf{w}$, θ and $\mathbf{z}$.

$k^* = k$.

for $j = 1$ to k^* **do**

  $\theta_j^* \sim p(\theta_j^*|\mathbf{z}, \mathbf{y}) \propto p(\theta_j^*|\gamma) \prod_{\{i:z_i^t=j\}} k(y_i|\theta_j^*)$.

  $v_j^* \sim p(v_j^*|\mathbf{z}, c) = \text{Be}\left(n_j + 1, n - \sum_{l=1}^{j} n_l + c\right)$.

  Calculate $\mathbf{w}^*$, i.e.

$$w_j^* = \begin{cases} v_1^* & \text{if } j = 1, \\ w_{j-1}^* \left(1 - v_{j-1}^*\right) \left(v_j^*/v_{j-1}^*\right) & \text{if } j \geq 2. \end{cases}$$

end for

for $i = 1$ to n **do**

  $u \sim \text{U}(0, 1)$. Let $j = 1$.

  while $\sum_{l=1}^{j} q(\mathbf{z}(i, l)|\mathbf{z}) \leq u$ **do**

    $j = j + 1$

    if $k^* < j$ **then**

      $k^* = k^* + 1$. {Update the number of groups (only if needed)}

      $v_j^* \sim \text{Be}(1, c)$ and $\theta_j^* \sim p(\theta_j^*|\gamma)$. {Generate the extra random variables}

      $w_j^* = w_{j-1}^*(1 - v_{j-1}^*)(v_j^*/v_{j-1}^*)$. {Generate the weights via (3.16)}

    end if

  end while

  Set $z_i = j$ with probability $\alpha(\mathbf{z}, \mathbf{z}(i, j))$.

  $k = \max\{\mathbf{z}\}$.

end for

return k, $\mathbf{z}$.

Remark 3.22. The key point under (3.39) is that if $\xi_1, \xi_2, \ldots$ is taken as a decreasing sequence, hence z_i is forced to be from a finite set.

To see this let us assume that $h : \mathbb{R} \to \mathbb{R}^+$ is a continuous decreasing function (thus invertible), i.e. for $x, z \in \mathbb{R}$, then

$$x < z \Leftrightarrow h(z) < h(x), \tag{3.40}$$

with inverse h^{-1}, and define a deterministic positive decreasing sequence $\xi_1, \xi_2, \ldots$, via $\xi_j = h(j)$, for $j = 1, \ldots$, hence from (3.39) and (3.40)

$$u_i \sim \mathrm{U}(u_i|0, h(z_i)) \quad \Leftrightarrow \quad u_i < h(z_i) \quad \Leftrightarrow \quad z_i < h^{-1}(u_i),$$

so if $k_i = \lfloor h^{-1}(u_i) \rfloor$ (where $\lfloor a \rfloor$ is the closest integer to a less or equal than a) it follows that

$$z_i \leq k_i \leq h^{-1}(u_i) \Rightarrow \; z_i \in \{1, \ldots, k_i\}, \text{ for } i = 1, \ldots, n.$$

Now z_i is chosen from a finite set, and we can devise a simple Gibbs algorithm to sample from (3.39). We display a summary of this strategy in Algorithm 3.9, this deals with Steps 7-10 of Algorithm 3.5.

Algorithm 3.9 Slice sampler

Require: $\mathbf{w}$, θ and (u_i, k_i), for $i = 1, \ldots, n$.

1: **for** $i = 1$ to n **do**

2: $z_i \sim p(z_i = j|u_i, \mathbf{w}, \theta, \mathbf{y}) = \dfrac{w_j k(y_i|\theta_j)}{\sum_{j=1}^{k_i} w_j k(y_i|\theta_j)} \quad (j = 1, \ldots, k_i).$

3: $u_i \sim p(u_i|z_i) = \mathrm{U}(u_i|0, h(z_i))$.

4: $k_i = \lfloor h^{-1}(u_i) \rfloor$.

5: **end for**

6: $k = \max_{i=1,\ldots,n} \{k_i\}$.

7: **return** k, $\mathbf{z}$ and (u_i, k_i) for $i = 1, \ldots, n$.

The choice of $\xi_1, \xi_2, \ldots$ is a delicate issue and any choice has to balance efficiency and computational time. Kalli, Griffin, and Walker (2011) found that the mixing depends

on the rate at which the ration $r_j = \mathbb{E}(w_j)/\xi_j$ increases with j. Faster rates of increase are associated with better mixing but longer running times, since the average size of the set $A_u = \{j : u > \xi_j\}$ increases. The authors suggested increasing the rate of increase of r_i until the gains in mixing are counter-balanced by the longer running time. They reported that $r_j \propto (1.5)^j$ strikes a good balance.

3.2.2.4 Label switching moves for infinite mixtures

Despite the weights of the Dirichlet process being weakly identifiable, see (3.9), the posterior distribution of each w_j is still multimodal. However, the sampler may get trapped in one of the local modes, thus, to improve mixing (escape trapping states and improve convergence) Papaspiliopoulos and Roberts (2008), included "label switching moves".

Remark 3.23. See Chapter 4 that is devoted to the label switching problem under a finite mixture setting, and Section 2.2.5, of Chapter 2, for a discussion on a related problem.

There are two complementary moves, each designed to force the appearance of the label switching phenomenon. The proposals are devised using the structure of the weights of the Dirichlet process. The first proposes to change the labels j and l of two randomly chosen components $(n_j, n_l > 0)$. The probability of such a change is accepted with probability

$$\alpha(j,l) = \min\left\{1, \left(\frac{w_j}{w_l}\right)^{n_l - n_j}\right\}. \tag{3.41}$$

This proposal will be highly accepted if the selected weights are similar. The second move proposes to change the labels j and $j+1$, and at the same time to exchange v_j and v_{j+1}. This change is accepted with probability

$$\alpha(j,j+1) = \min\left\{1, \frac{(1-v_{j+1})^{n_j}}{(1-v_j)^{n_{j+1}}}\right\}. \tag{3.42}$$

The probability of acceptance for this proposal is high if the clusters are very unequal.

3.2.2.5 Additional comments

- **Inference about the number of components**. In the MCMC sampler for the finite mixture model with k components (k known or unknown), see Algorithm 2.1 or Section 2.3.2, a certain number of empty components are generated at each iteration of the sampler. These are components who have no observations assigned to them. This happens despite the fact that at each iteration of the sampler the number of choices for z_i is the same, k, for $i = 1, \ldots, n$, and that given $\delta = 1$, the prior for the weights is uniform over the space $w_1 + \cdots + w_k = 1$. For the infinite mixture model, each z_i has k_i choices, for $i = 1, \ldots, n$. This can clearly be seen in the slice sampler strategy, but it is also true in the retrospective sampler, and the prior for the weights is rather disperse. This generates large numbers of empty components throughout iterations of the MCMC sampler, and thus clearly inflating the estimated number of underlying groups or components k in the sample. A common solution is to store, at each iteration of the MCMC strategy, the number of non-empty components, let us say k_{full}, where $k = k_{full} + k_{empty}$, and k_{empty} is the number of empty components. Then, base the inference for the number of groups only in k_{full}. Note that following this idea it is also possible to make inference for the number of components via the truncation method.

- **Comparative between retrospective and marginal methods**. To compare the performance between the retrospective and conditional samplers, Papaspiliopoulos and Roberts (2008) performed a simulation study. The measure of comparison was the integrated auto-correlation time. Integrated auto-correlation time is of interest as it controls the variance of the Monte Carlo estimator $\bar{f}$ of f, thus if all the estimators have converged to f, the estimator with the smallest integrated auto-correlation time should be the most accurate. These comparisons showed that marginal methods produce smaller integrated auto-correlation times.

After a careful inspection Papaspiliopoulos and Roberts (2008) suggested that this may be due to the fact that the retrospective sampler has to explore multiple modes in the posterior distribution; On the other hand, the labelling is not important in the marginal approaches, which work with the unidentifiable allocation structure and do not need to explore a multimodal distribution. Finally, they observed that the label-switching moves substantially improved the performance of their algorithm.

- **Comparative between conditional methods**. A comparison between conditional methods was carried out by Kalli, Griffin, and Walker (2011). Here again the measure of comparison was the integrated auto-correlation time. The comparison was carried out using different alternatives for $\xi_1, \xi_2, \ldots$, for the slice sampler, and choosing different levels or error for the truncation method. In the end the difference between the truncation method, retrospective sampler and the most efficient slice sampler was marginal, however it was found that the retrospective sampler produced the smallest auto-correlation error. It is important to say that comparing the retrospective and slice sampler, Algorithms 3.8 and 3.9, respectively, is easy to see that the slice sampler is much easier to implement than the retrospective algorithm.

Chapter 4

Label Switching in Finite Mixture Models

As described in Chapter 1, label switching is a well known problem in the Bayesian analysis of finite mixture models. On the one hand, it has been perceived as a prerequisite to justify MCMC convergence and on the other hand it complicates inference. In this Chapter we review the strategies to force the appearance of label switching, and then focus our attention on the proposals to undo the label switching deterministically.

We first describe a general framework in which deterministic relabeling algorithms can be expressed as versions of the assignment problem: with different costs, to then formulate a new relabeling algorithm following these ideas. Finally, we compare our proposal with previous relabeling algorithms on univariate and multivariate data examples.

In this Chapter we will assume the number of components k to be known.

4.1 Label switching in mixture models

The ultimate objective under any relabeling algorithm is to gain identifiability for the components, and then to be able to perform component specific inference.

4.1.1 Component specific inference in mixture models

The latent variables (2.2) introduce a natural clustering structure over the observations and are the basis for making inference about the hidden groups. Under this framework, the goal is to find the cluster from which each y_i is drawn. Hence, to group a set of observations into k clusters, we make inference about the classification probabilities of each observation. If a single best clustering is all that is needed, each observation is assigned to the cluster with the highest classification probability. In the rare case that the parameters of the mixture are known, Bayes' Theorem can be used to calculate the classification probabilities directly:

$$\begin{aligned} p(z_i = j|y_i, \mathbf{w}, \boldsymbol{\theta}) &= \frac{p(z_i = j|\mathbf{w})p(y_i|\boldsymbol{\theta}, z_i = j)}{\sum_{l=1}^{k} p(z_i = l|\mathbf{w})p(y_i|\boldsymbol{\theta}, z_i = l)}, \\ &= \frac{w_j f(y_i|\theta_j)}{\sum_{l=1}^{k} w_l f(y_i|\theta_l)}. \end{aligned} \tag{4.1}$$

When the parameters of the mixture are unknown, we need to calculate the posterior classification probabilities:

$$\begin{aligned} p(z_i = j|\mathbf{y}) &= \int_{\boldsymbol{\Theta}} \int_{\boldsymbol{\Omega}} p(z_i = j, \mathbf{w}, \boldsymbol{\theta}|\mathbf{y}) \mathbf{dw} d\boldsymbol{\theta}, \\ &= \int_{\boldsymbol{\Theta}} \int_{\boldsymbol{\Omega}} p(z_i = j|\mathbf{w}, \boldsymbol{\theta}, y_i) p(\mathbf{w}, \boldsymbol{\theta}|\mathbf{y}) \mathbf{dw} d\boldsymbol{\theta}, \\ &= \int_{\boldsymbol{\Theta}} \int_{\boldsymbol{\Omega}} \frac{w_j f(y_i|\theta_j)}{\sum_{l=1}^{k} w_l f(y_i|\theta_l)} p(\mathbf{w}, \boldsymbol{\theta}|\mathbf{y}) \mathbf{dw} d\boldsymbol{\theta}. \end{aligned} \tag{4.2}$$

where $\mathbf{y} = \{y_i\}_{i=1}^{n}$. Using MCMC methods, (4.2) can be approximated via:

$$p(z_i = j|\mathbf{y}) \approx \frac{1}{N} \sum_{t=1}^{N} \frac{w_j^t f(y_i|\theta_j^t)}{\sum_{l=1}^{k} w_l^t f(y_i|\theta_l^t)}, \text{(for } N \text{ large enough)} \tag{4.3}$$

where $(\mathbf{w}^t, \boldsymbol{\theta}^t) \sim p(\mathbf{w}, \boldsymbol{\theta}|\mathbf{y})$ for $t = 1, \ldots, N$ (t indexes the iterations of the MCMC).

To classify a new observation y^*, with y^* conditionally independent of $\mathbf{y}$, we calculate

$$\begin{aligned} p(z^* = j|y^*, \mathbf{y}) &= \int_{\boldsymbol{\Theta}} \int_{\boldsymbol{\Omega}} p(z^* = j, \mathbf{w}, \boldsymbol{\theta}|y^*, \mathbf{y}) \mathbf{dw} d\boldsymbol{\theta}, \\ &\propto \int_{\boldsymbol{\Theta}} \int_{\boldsymbol{\Omega}} p(z^* = j|\mathbf{w}, \boldsymbol{\theta}) p(y^*|\mathbf{w}, \boldsymbol{\theta}) p(\mathbf{y}|\mathbf{w}, \boldsymbol{\theta}) p(\mathbf{w}, \boldsymbol{\theta}) \mathbf{dw} d\boldsymbol{\theta}, \\ &\propto \int_{\boldsymbol{\Theta}} \int_{\boldsymbol{\Omega}} w_j f(y^*|\theta_j) p(\mathbf{w}, \boldsymbol{\theta}|\mathbf{y}) \mathbf{dw} d\boldsymbol{\theta}. \end{aligned} \tag{4.4}$$

This can be estimated in an analogous way to (4.3), with obvious changes. Note that (4.4) tells us that to make inference for the scaled predictive distribution of y^* is equivalent to the estimation of its classification probabilities. Finally, with the same MCMC output, posterior mean estimates for components' parameters (or functions of them) can be approximated by averaging over the parameters identified by the index of interest:

$$\mathbb{E}\left(h\left(w_j,\theta_j\right)|\mathbf{y}\right)=\int_{\boldsymbol{\Theta}}\int_{\boldsymbol{\Omega}}h\left(w_j,\theta_j\right)p(\mathbf{w},\boldsymbol{\theta}|\mathbf{y})\mathbf{dwd}\boldsymbol{\theta}\approx\frac{1}{N}\sum_{t=1}^{N}h\left(w_j^t,\theta_j^t\right). \tag{4.5}$$

Later on in the thesis our aim will be to estimate classification probabilities (4.2), scale densities and posterior means (4.5).

4.1.2 The label switching problem: a complication for inference

The likelihood of a mixture model is invariant under permutations of its parameters: let $\mathcal{P}_k$ be the set of the $k!$ permutations of the labels $\{1,\ldots,k\}$. If for some $\rho \in \mathcal{P}_k$ we define $\rho(\mathbf{w},\boldsymbol{\theta}):=((w_{\rho(1)},\ldots,w_{\rho(k)}),(\theta_{\rho(1)},\ldots,\theta_{\rho(k)}))$, then, for every $\rho,\nu \in \mathcal{P}_k$

$$p(\mathbf{y}|\rho(\mathbf{w},\boldsymbol{\theta}))=\prod_{i=1}^{n}\left[\sum_{j=1}^{k}w_{\rho(j)}f(y_i|\theta_{\rho(j)})\right]=\prod_{i=1}^{n}\left[\sum_{j=1}^{k}w_{\nu(j)}f(y_i|\theta_{\nu(j)})\right]=p(\mathbf{y}|\nu(\mathbf{w},\boldsymbol{\theta})).$$

Consequently, if the support of the parameters is the same, e.g. under symmetric priors across the components, the posterior distribution will inherit the likelihood's invariance. Hence, in an MCMC algorithm, the indices of the parameters can permute multiple times between iterations. As a result, we cannot identify the hidden groups which make (4.3), and all other ergodic averages to estimate characteristics of the components useless.

A popular idea when working with mixture models is to include the latent allocation variables (2.2). Note that, in this case, for every $\rho,\nu \in \mathcal{P}_k$

$$p(\mathbf{y}|\rho(\boldsymbol{\theta},\mathbf{z}))=\prod_{i=1}^{n}f(y_i|\theta_{\rho(z_i)})=\prod_{i=1}^{n}f(y_i|\theta_{\nu(z_i)})=p(\mathbf{y}|\nu(\boldsymbol{\theta},\mathbf{z})) \tag{4.6}$$

where for some $\rho \in \mathcal{P}_k$, $\rho(\boldsymbol{\theta},\mathbf{z}):=((\theta_{\rho(1)},\ldots,\theta_{\rho(k)}),(\rho(z_1),\ldots,\rho(z_n)))$. Hence, given the allocations $\mathbf{z}$, the likelihood (4.6) is invariant under permutations of the parameters as well.

4.1.3 MCMC convergence in mixture models

Under a mixture model set-up, to asses convergence of an MCMC strategy, marginal posterior estimates are monitored. For the Gibbs sampler, Algorithm 2.1, plots of estimates show that in some cases the labels of the components do not permute, see pp. 960 and 962 of Celeux, Hurn, and Robert (2000) and p. 55 of Jasra, Holmes, and Stephens (2005). This has been perceived as a symptom of poor mixing of the sampler. See, for example, Section 2.2.5.

To solve the lack of label switching, Frühwirth-Schnatter (2001) added a Metropolis-Hastings move that proposes a random permutation of the labels, which is accepted with probability one (the likelihood is invariant to permutation of the mixture parameters and we are working under symmetric priors). This solution can be implemented in a post-processing stage, without further modification of the sampler. However, it was discarded by Celeux, Hurn, and Robert (2000) p. 959. "Our insistence on searching for an algorithm that can achieve symmetry without such a move is that any concerns over convergence are not necessarily dealt by such a strategy, which simply alleviates the most obvious symptom", and in Jasra, Holmes, and Stephens (2005) p. 55. it is stated "this course of action is only appropriate if the posterior distribution is not genuinely multimodal (which would not be known a priori to simulation)". Both sets of authors concluded that alternative and more complex MCMC samplers should be considered. A broader discussion along the same lines can be found in Geweke (2007), who is not sympathetic to the construction of complex MCMC algorithms.

To address the convergence problem, Papastamoulis and Iliopoulos (2010) borrowed the ideas of Papaspiliopoulos and Roberts (2008) and implemented at each iteration of a Gibbs sampler, a Metropolis-Hastings step that proposes a random permutation between two labels of the mixture: $j, l \in \{1, \ldots, k\}$. The acceptance probability for this proposal is given by (3.41). However, this idea was designed to introduce label switching in infinite mixtures, and under this setting the weights are weakly identifiable, see remark 3.9 in

Chapter 3. This is not the case for the weights of a finite mixture model, so this proposal is an alternative to the ideas described in Frühwirth-Schnatter (2001).

The last alternative is to use a trans-dimensional sampler. These samplers improve mixing within k, see Section 2.4. Note that this solution can be computationally demanding and may be inefficient, due to the fact that the chain will not necessarily stay in the model of interest for the entire run, see Richardson and Green (1997), Stephens (1997) (Section 4.4.1) and Jasra, Holmes, and Stephens (2005).

A further justification for complex MCMC and trans-dimensional samplers is the possibility of genuine multimodality. Genuine multimodality is more than the modes being identical up to permutation of the labels; see Sections 2.3 and 3.4.2 of Stephens (1997). Hence, the argument against the Gibbs sampler is that in the presence of genuine multimodality, it can get trapped in one of the symmetric modes, and not visiting genuine modes.

4.2 Deterministic relabeling strategies

Suppose a sample from the posterior distribution of (2.1) is generated: $(\mathbf{w}^t, \boldsymbol{\theta}^t) \sim p(\mathbf{w}, \boldsymbol{\theta}|\mathbf{y})$ for $t = 1, \ldots, N$. The aim under any relabeling strategy is to find a permutation $\rho_t \in \mathcal{P}_k$, such that the permuted sample:

$$\rho_t(\mathbf{w}^t, \boldsymbol{\theta}^t) = ((w^t_{\rho_t(1)}, \ldots, w^t_{\rho_t(k)}), (\theta^t_{\rho_t(1)}, \ldots, \theta^t_{\rho_t(k)})), \tag{4.7}$$

for $t = 1, \ldots, N$, recovers identifiably for the mixture components.

An obvious way to proceed is to find artificial constraints over the parameter space of $(\mathbf{w}, \boldsymbol{\theta})$. It is important to note that, in this case, the permutation $\rho_t \in \mathcal{P}_k$ will be the one for which the constraints are satisfied. We can start with basic constraints and then use summaries of the MCMC output to update them; see Frühwirth-Schnatter (2001). While this might work in the univariate case, it is challenging to extend to the multivariate setting. Thus, the objective is to find constraints automatically via

deterministic algorithms. This is reduced to finding the permutation ρ_t that minimizes a loss function. It is important to stress that such a strategy can be implemented as an on-line method or post-processing the sample, without modifying the MCMC, hence, allowing us to make inference with the permuted MCMC sample. For a formal justification, see Stephens (1997), Proposition 3.1.

The key points of any deterministic relabeling strategy are the definition of the loss function, the selection of a pivot and the minimization step. Let $\mathcal{L}^t$ be the loss function to minimize, and $c^t_{l,j}$ be the cost associated if permutation $\rho_t(l) = j$ is chosen. The minimization step for the sampled values at iteration t of the MCMC can be formulated via the well known assignment problem, see Burkard, Dell'Amico, and Martello (2009):

$$\begin{aligned} \text{Minimize} \quad & \mathcal{L}^t = \sum_{l=1}^{k}\sum_{j=1}^{k} a_{l,j} c^t_{l,j} \\ \text{subject to} \quad & \sum_{j=1}^{k} a_{l,j} = \sum_{l=1}^{k} a_{l,j} = 1 \quad (l, j = 1, \ldots, k), \\ & a_{l,j} \in \{0, 1\} \qquad\quad (l, j = 1, \ldots, k). \end{aligned} \tag{4.8}$$

We need to solve (4.8) by minimizing over the $(a_{l,j})$; if for some t, $(\widehat{a}_{l,j})$ is an optimal solution for the problem and it is that $\widehat{a}_{l,j} = 1$, then this corresponds to the permutation $\rho_t(l) = j$. Also, let $\widehat{\mathcal{L}}^t$ be the loss function (4.8) evaluated at $(\widehat{a}_{l,j})$. This formulation was first used by Stephens (1997). In the next Section we describe the algorithms, loss functions and pivots used by Stephens (2000b) and Papastamoulis and Iliopoulos (2010).

4.2.1 Kullback-Leibler relabeling

The idea behind Stephens' Kullback-Leibler (KL) relabeling is straightforward; see Stephens (2000b). First, during the MCMC algorithm the classification probabilities are stored. Let $\mathbf{P}^t_{n\times k}$ be the matrix of classification probabilities (at iteration t of the MCMC), i.e.

$$\mathbf{P}^t = \{\{p^t_{i,j}\}^k_{j=1}\}^n_{i=1} \quad \text{with} \quad p^t_{i,j} = \frac{w^t_j f(y_i|\theta^t_j)}{\sum_{l=1}^k w^t_l f(y_i|\theta^t_l)}.$$

Second, let $\mathbf{Q}_{n\times k}$ be the matrix of "true classification probabilities". Under Stephens' setting the matrix $\mathbf{Q}$ is the pivot, and the goal is to find a permutation $\rho_t \in \mathcal{P}_k$, of the columns of $\mathbf{P}^t$: $\rho_t(\mathbf{P}^t) := \{\{p^t_{i,\rho_t(j)}\}_{j=1}^k\}_{i=1}^n$, such that the Kullback-Leibler divergence between $\rho_t(\mathbf{P}^t)$ and $\mathbf{Q}$ is minimum over all permutations. Clearly, $\mathbf{Q}$ is not known, thus it is approximated via a recursive algorithm. We provide a description of this strategy in Algorithm 4.10.

Algorithm 4.10 KL relabeling strategy

1: Initialize $\rho_1, \ldots, \rho_N$. {Usually set to the identity: $\rho_t = \{1, \ldots, k\}$ for all t}

2: For $i = 1$ to n and $j = 1$ to k, calculate $\widehat{q}_{i,j} = \dfrac{1}{N}\displaystyle\sum_{t=1}^{N} p^t_{i,\rho_t(j)}$. {estimating $\mathbf{Q}$}

3: For $t = 1$ to N, find ρ_t solving (4.8) with costs: $c^t_{l,j} = \displaystyle\sum_{i=1}^{n} p^t_{i,j} \log\left(\frac{p^t_{i,j}}{\widehat{q}_{i,l}}\right)$.

4: If an improvement in $\sum_{t=1}^{N} \widehat{\mathcal{L}}^t$ has been achieved return to 2. Finish otherwise.

4.2.2 Equivalent classes representatives relabeling

Equivalent classes representatives (ECR) method, Papastamoulis and Iliopoulos (2010), is a fast and easy strategy to undo the label switching. The idea is as follows. We can define two allocations $\mathbf{z^1}$ and $\mathbf{z^2}$ as equivalent if there is a permutation $\rho \in \mathcal{P}_k$ such that $\mathbf{z^1} = \rho(\mathbf{z^2}) = \{\rho(z^2_1), \ldots, \rho(z^2_n)\}$. Hence, if we let $\mathbf{g} = \{g_i\}_{i=1}^n$ be the vector of "true allocations" and $\mathbf{z}^t$ the vector of allocations being sampled at iteration t of the MCMC algorithm, to undo the label switching, we find the permutation $\rho_t \in \mathcal{P}_k$ that minimizes the simple matching distance (SMD) between $\mathbf{g}$ and $\rho_t(\mathbf{z}^t) = (\rho_t(z^t_1), \ldots, \rho_t(z^t_n))$, for $t = 1, \ldots, N$:

$$\mathrm{SMD}(\mathbf{g}, \rho_t(\mathbf{z}^t)) = n - \sum_{i=1}^{n} \mathbb{1}\{g_i = \rho_t(z^t_i)\}.$$

Thus, in this case, the pivot will be the vector $\mathbf{g}$. An easy solution that avoids a recursive algorithm to estimate $\mathbf{g}$ is to use the MCMC output to choose a "good" allocation vector, and following Martin, Mengersen, and Robert (2005), Papastamoulis and Iliopoulos (2010) used the maximum aposteriori estimator (MAP): at each iteration

of the MCMC we store the latent allocations, $\mathbf{z}^t$, and the log posterior distribution, $\ell^t = \log\{p(\mathbf{w}^t, \theta^t, \mathbf{z}^t|\mathbf{y})\}$. To choose the MAP estimate, we find t^* such that $\ell^{t^*} = \max_{t=1,\ldots,N} \ell^t$ and let $\mathbf{g}^{\mathbf{MAP}} = \mathbf{z}^{t^*}$. A description of the ECR relabeling strategy is provided in Algorithm 4.11: note that $n_j^t = \#\{i : z_i^t = j\}$ and $n_{j,l}^t = \#\{i : z_i^t = j, g_i^{\mathbf{MAP}} = l\}$.

Algorithm 4.11 ECR relabeling strategy

1: Find $\mathbf{g}^{\mathbf{MAP}}$.

2: For $t = 1$ to N, find ρ_t solving (4.8) with costs: $c_{l,j}^t = n_j^t - n_{j,l}^t$.

From Algorithm 4.11 it is not clear how the costs work. We have devised a simple example to understand this point: let $\mathbf{z}^t$ and $\mathbf{g}^{\mathbf{MAP}}$ be as in Table 4.1. It is clear the permutation that minimizes the SMD is $\rho_t = \{3, 1, 2, 4\}$ hence, $\rho_t(1) = 3$, $\rho_t(2) = 1$, $\rho_t(3) = 2$ and $\rho_t(4) = 4$. Then, the costs are matching distances translated as comparisons of counts.

i	1	2	3	4	5	6	7	8	9	10	11	12
$\mathbf{z}^t$	2	2	2	3	3	3	3	1	1	1	1	4
$\mathbf{g}^{\mathbf{MAP}}$	1	1	1	1	2	2	2	2	3	3	3	4

Table 4.1: Example ECR, $\mathbf{z}^t$ and $\mathbf{g}^{\mathbf{MAP}}$

According to Papastamoulis and Iliopoulos (2010), besides the MAP estimate, other valid choices for good allocation vectors are the most probable allocation and the allocation vector corresponding to the maximum of the likelihood (4.6).

We stress that in this summary of the ECR algorithm we have restricted our attention to the costs. However, to show the sampler converges to the correct posterior distribution, Papastamoulis and Iliopoulos (2010) gave more emphasis to arguments about the equivalent classes induced by $\rho(\mathbf{z})$ for $\rho \in \mathcal{P}_k$ and convergence proofs: the name "representatives" comes from taking a set $\mathcal{Z}_0$ consisting of exactly one representative from each equivalent class and then showing that a sample from the posterior $p(\mathbf{w}, \boldsymbol{\theta}|\mathbf{y})$ can be obtained by calculating the posterior restricted to $\mathcal{Z}_0$ and then permuting the labels

of the simulated values.

4.2.3 Comments on KL and ECR algorithms

The KL and ECR algorithms need estimated pivot variables $\widehat{\mathbf{Q}}$ and $\widehat{\mathbf{g}}$, respectively. We stress that these must be chosen as close as possible *to the mode* of one of the $k!$ symmetric modes of the posterior distribution: the aim is to eliminate the influence of the remaining symmetric copies. The KL algorithm uses a posterior mean and the ECR strategy aims for a posterior mode. This is an important point because it is valid for all relabeling algorithms. KL achieves its pivot starting from a bad estimate and improving it via fixed point theory ideas, thus leading to an iterative algorithm that converges to a local minimum. While ECR aims to find a good estimate from the start (usually the MAP estimate). Once the pivot variables are obtained, these are used to permute the MCMC output, i.e. (4.7), and obtain a representative of one of the modes of the posterior distribution.

The real difference between the algorithms are the costs used in the loss function (4.8). It is possible to use the MAP estimator of $\mathbf{Q}$: $\mathbf{Q}^{\mathbf{MAP}}$ on the KL algorithm. In the same manner, via Algorithm 1 of Stephens (1997) p. 800, an estimate; $\widehat{\mathbf{g}} = \{\widehat{g}_i\}_{i=1}^n$ of $\mathbf{g}$ can be obtained. A description of the ECR strategy via an iterative process is provided in Algorithm 4.12, and in this case $n_{j,l}^t = \#\{i : z_i^t = j, \widehat{g}_i = l\}$. Now we start with a bad estimate $\widehat{\mathbf{g}}$ and improve it via an iterative algorithm. The last iteration will be exactly the original ECR algorithm but where the pivot was obtained in an alternative manner.

Algorithm 4.12 ECR iterative relabeling strategy 1

1: Initialize $\rho_1, \ldots, \rho_N$. {Usually set to the identity: $\rho_t = \{1, \ldots, k\}$ for all t}

2: For $i = 1$ to n, calculate $\widehat{g}_i = \text{mode}\{\rho_1(z_i^1), \ldots, \rho_N(z_i^N)\}$. {estimating $\mathbf{g}$}

3: For $t = 1$ to N, find ρ_t solving (4.8) with costs: $c_{l,j}^t = n_j^t - n_{j,l}^t$.

4: If an improvement in $\sum_{t=1}^N \widehat{\mathcal{L}}^t$ has been achieved return to 2. Finish otherwise.

Note however, that neither in the original ECR nor in the iterative version the pivot

$\widehat{\mathbf{g}}$ will coincide with the final estimate for the single best clustering. An alternative that will ensure this is provided in Algorithm 4.13. This algorithm will be computationally demanding but will give the permutations, the single best clustering and the posterior classification probabilities as an output.

Algorithm 4.13 ECR iterative relabeling strategy 2

1: Initialize $\rho_1, \ldots, \rho_N$. {Usually set to the identity: $\rho_t = \{1, \ldots, k\}$ for all t}

2: For $i = 1$ to n and $j = 1$ to k, calculate $\widehat{q}_{i,j} = \dfrac{1}{N} \displaystyle\sum_{t=1}^{N} p^t_{i,\rho_t(j)}$.

3: For $i = 1$ to n, set $\widehat{g}_i = j$ if $\widehat{q}_{i,j} = \max\limits_{l=1\ldots,k} \{\widehat{q}_{i,l}\}$. {estimating $\mathbf{g}$}

4: For $t = 1$ to N, find ρ_t solving (4.8) with costs: $c^t_{l,j} = n^t_j - n^t_{j,l}$.

5: If an improvement in $\sum_{t=1}^{N} \widehat{\mathcal{L}}^t$ has been achieved return to 2. Finish otherwise.

4.3 Using the data to undo the label switching

Our aim is to combine all the information about the clusters gathered by the latent allocations, (2.2), and the data, to devise a simple loss function. Then, as with other relabeling algorithms, to undo the label switching, we will find the permutation of the indices that minimizes a loss.

The key point is that if the MCMC has converged, from iteration to iteration, the labels of each cluster may change. But the clusters should be roughly the same. The difference should be small enough for us to discover where a cluster of the data has moved to. Hence, all that we need is to keep track of the k clusters throughout each iteration of the sampler. For now, to explain our ideas, we will restrict our attention to the univariate case. The multivariate problem is a trivial generalization and is explained in the illustrations for multivariate data.

4.3.1 A simple loss function

Let us assume that the center and the dispersion of the clusters within the data are known: $\{\mathsf{m}_1, \ldots, \mathsf{m}_k\}$ and $\{\mathsf{s}_1, \ldots, \mathsf{s}_k\}$. These will be our pivot variables. To undo the label switching we can find the permutation, $\rho_t \in \mathcal{P}_k$ for $t = 1, \ldots, N$, that minimizes:

$$\sum_{l=1}^{k} \sum_{j=1}^{k} n^t_{\rho_t(j)} \left\{ \sum_{\{i:\rho_t(z_i^t)=j\}} \left(\frac{y_i - \mathsf{m}_l}{\mathsf{s}_l} \right)^2 \right\}, \tag{4.9}$$

where $n^t_{\rho_t(j)} = \#\{i : \rho_t(z_i^t) = j\}$. With this loss the costs in (4.8) will be given by

$$c_{l,j} = n_j \sum_{\{i:z_i=j\}} \left(\frac{y_i - \mathsf{m}_l}{\mathsf{s}_l} \right)^2. \tag{4.10}$$

If the clustering assumptions induced by (2.1) and (2.2) are met, once the MCMC has converged, at each iteration of the sampler there must be k clusters identified by the conditional relationship between y_i and z_i^t. Thus, a k-means type of diverging measure, (4.9), will be able to keep track of each cluster. To measure deviations from the center of each cluster, m_l, we eliminate the effect of the scale, and as the number of observations allocated within each cluster can be rather distinct, we make the divergences proportional to the size of each group. These considerations improved the loss function greatly.

4.3.1.1 Means and standard deviations

There are different alternatives for measures of central tendency and dispersion for the clusters, and here we derive algorithms based on the mean and standard deviation. Later on in this Chapter we describe how to use more robust alternatives.

To estimate the means and standard deviations for each cluster we could use the mixture parameters or functions of them, for example, in the case of the normal mixture the obvious options are posterior means for the means and standard deviations of each component, i.e. $\widehat{\mathsf{m}}_j = \mathbb{E}(\mu_j|\mathbf{y})$ and $\widehat{\mathsf{s}}_j = \mathbb{E}(\sigma_j|\mathbf{y})$, for $j = 1, \ldots, k$. However, there are two problems with this approach. First, the algorithm would be restricted to mixtures with a particular component's distribution. Second, we would need to estimate these

posterior means from the MCMC output, and the parameters sampled via the MCMC are random and can exhibit great amounts of variability depending on whether the Markov chain traverses regions of high or low probability in the posterior distribution. To solve these problems we will work with functions depending only on the data and the latent allocations. The data is fixed, and there is only certain amount of variability that subsets of it will admit. On the other hand, the distributional assumptions of the model are contained in the clusters induced by the latent allocations.

With these ideas, to estimate the means and standard deviations of each cluster we use posterior means for the sample mean and sample standard deviation:

$$\widehat{\mathsf{m}}_j = \mathbb{E}(\overline{y}_j|\mathbf{y}) \text{ and } \widehat{\mathsf{s}}_j = \mathbb{E}(s_j|\mathbf{y}), \text{ for } j = 1, \ldots, k, \tag{4.11}$$

where $\overline{y}_j = \dfrac{1}{n_j} \sum_{\{i:z_i=j\}} y_i$ and $s_j^2 = \dfrac{1}{n_j - 1} \sum_{\{i:z_i=j\}} (y_i - \overline{y}_j)^2$.

Observe that quantities such as (4.11) are valid posterior means. To see this first let $\mathcal{Z} = \{1, \ldots, k\}^n$, and then for $h : \mathbb{R}^n \times \mathcal{Z} \times \{1, \ldots, k\} \to \mathbb{R}$, calculate

$$\begin{aligned} \mathbb{E}(h(\mathbf{y}, \mathbf{z}, j)|\mathbf{y}) &= \sum_{\mathbf{z}\in\mathcal{Z}} \int_{\boldsymbol{\Theta}} \int_{\boldsymbol{\Omega}} h(\mathbf{y}, \mathbf{z}, j) p(\mathbf{z}, \mathbf{w}, \boldsymbol{\theta}|\mathbf{y}) \mathbf{dw} \mathbf{d}\boldsymbol{\theta}, \\ &= \sum_{\mathbf{z}\in\mathcal{Z}} h(\mathbf{y}, \mathbf{z}, j) p(\mathbf{z}|\mathbf{y}). \end{aligned} \tag{4.12}$$

Thus, for (4.11), the function $h(\mathbf{y}, \mathbf{z}, j)$ will be given by $\overline{y}_j$ and s_j for the sample mean and sample standard deviation, respectively.

The posterior means in (4.11), will be estimated using the MCMC output and calculating:

$$\overline{y}_j^t = \frac{1}{n_j^t} \sum_{\{i:z_i^t=j\}} y_i \text{ and } {s_j}^t = \left(\frac{1}{n_j^t - 1} \sum_{\{i:z_i^t=j\}} (y_i - \overline{y}_j^t)^2 \right)^{1/2}.$$

Next section, specifically Algorithm 4.14 explains the technical details of $n_j^t = 0$ or 1.

4.3.2 The algorithm

As with the previous algorithms, to define the costs (4.10), a complete knowledge about the pivot variables is assumed, in our case, the means and standard deviations of each

group. We have two ways to proceed: via a recursive algorithm or using estimates. We use the second approach simply because it is faster, but for a comprehensive explanation we include both.

The idea to find estimates is as follows: first, we obtain initial estimates of our pivot variables in one of the modes of the posterior distribution, in this case (4.11). If the MCMC has converged, we choose the first set of allocations after the burn in period and calculate our starting sample means and sample standard deviations with it. At this point the MCMC should be traversing one of the $k!$ modes of the posterior distribution, and for our propose it is not important which particular mode this is. To move these initial pivots closer to *the mode* of this particular mode of the posterior distribution, we average the same estimates for each group over all the iterations of the MCMC algorithm: preserving identifiability via (4.9). The final estimates will be close to *the mode* of one of the modes of the posterior distribution, and we can use these as pivots to find the permutations (4.7). A formal description of the strategy to find estimates is given in Algorithm 4.14 (note that $R = y_{(n)} - y_{(1)}$).

Algorithm 4.14 Data based relabeling: finding estimates

1: For $j = 1$ to k, initialize $n_j^{\mathsf{m}} = 1$, $n_j^{\mathsf{s}} = 1$, $\widehat{\mathsf{m}}_j = y_{(1)} + R\left(\frac{j}{k+1}\right)$ and $\widehat{\mathsf{s}}_j = \sqrt{2}R/k$.

2: For $t = 1$ to N, find ρ_t solving (4.8) with costs: $c_{l,j}^t = n_j^t \sum_{\{i:z_i^t=j\}} \left(\frac{y_i - \widehat{\mathsf{m}}_l}{\widehat{\mathsf{s}}_l}\right)^2$.

For $j = 1$ to k, update $\widehat{\mathsf{m}}_j$, $\widehat{\mathsf{s}}_j$, n_j^{m} and n_j^{s}

$$\text{If } n_{\rho_t(j)}^t > 0 \Rightarrow \widehat{\mathsf{m}}_j = ((n_j^{\mathsf{m}} - 1)\widehat{\mathsf{m}}_j + \overline{y}_{\rho_t(j)}^t)/n_j^{\mathsf{m}} \text{ and } n_j^{\mathsf{m}} = n_j^{\mathsf{m}} + 1.$$

$$\text{If } n_{\rho_t(j)}^t > 1 \Rightarrow \widehat{\mathsf{s}}_j = ((n_j^{\mathsf{s}} - 1)\widehat{\mathsf{s}}_j + s_{\rho_t(j)}^t)/n_j^{\mathsf{s}} \text{ and } n_j^{\mathsf{s}} = n_j^{\mathsf{s}} + 1.$$

A description of the Data relabeling via estimates is provided in Algorithm 4.15. We have defined a two step procedure: in Algorithm 4.14 the aim is to find estimates for the mean and standard deviation for each cluster. While in Algorithm 4.15 the interest lies in the permutations ρ_t, for $t = 1, \ldots, N$. It is important to say that several experiments

were performed using Algorithm 4.14, to obtain the permutations directly, and in some cases the results were as good as with the two step procedure. However, in general the two step procedure was more accurate.

Algorithm 4.15 Data based relabeling: estimates strategy

1: Find estimates $\widehat{\mathsf{m}}_j$ and $\widehat{\mathsf{s}}_j$ for $j = 1$ to k.

2: For $t = 1$ to N, find ρ_t solving (4.8) with costs: $c^t_{l,j} = n^t_j \sum_{\{i:z^t_i=j\}} \left(\frac{y_i - \widehat{\mathsf{m}}_l}{\widehat{\mathsf{s}}_l} \right)^2$.

The corresponding recursive algorithm follows directly from Algorithm 1 of Stephens (1997), p. 800. A description of this strategy is given in Algorithm 4.16.

Algorithm 4.16 Data based relabeling: iterative strategy

1: Initialize $\rho_1, \ldots, \rho_N$. {Usually set to the identity: $\rho_t = \{1, \ldots, k\}$ for all t}

2: For $j = 1$ to k, calculate:

$$\widehat{\mathsf{m}}_j = \frac{1}{n^{\mathsf{m}}} \sum_{t=1}^{N} \overline{y}^t_{\rho_t(j)} \, \mathbb{1}\{n^t_{\rho_t(j)} > 0\} \text{ and } \widehat{\mathsf{s}}_j = \frac{1}{n^{\mathsf{s}}} \sum_{t=1}^{N} s^t_{\rho_t(j)} \, \mathbb{1}\{n^t_{\rho_t(j)} > 1\},$$

where $n^{\mathsf{m}} = \sum_{t=1}^{N} \mathbb{1}\{n^t_{\rho_t(j)} > 0\}$ and $n^{\mathsf{s}} = \sum_{t=1}^{N} \mathbb{1}\{n^t_{\rho_t(j)} > 1\}$.

3: For $t = 1$ to N, find ρ_t solving (4.8) with costs: $c^t_{l,j} = n^t_j \sum_{\{i:z^t_i=j\}} \left(\frac{y_i - \widehat{\mathsf{m}}_l}{\widehat{\mathsf{s}}_l} \right)^2$.

4: If an improvement in $\sum_{t=1}^{N} \widehat{\mathcal{L}}^t$ has been achieved return to 2. Finish otherwise.

4.3.3 Additional comments

In this section we describe the ideas under the definition of the costs (4.10) and possible alternatives.

4.3.3.1 Main ideas

There are two important ideas under the definition of (4.10). First, it is easier to obtain estimates (pivots) close to *the mode* of one of the modes of the posterior distribution using general characteristics of the components such as location and dispersion, rather

than via those related to individual observations such as classification probabilities or allocations. If the algorithm has converged, we can obtain estimates for the location and dispersion for the groups directly from the MCMC output: the relationship between the observations and allocations is the key. Second, the observations are incorporated directly within the loss, and not only via the MCMC output. Indeed, note that all the quantities needed for the calculation of the costs, (4.10), are based on the data, while the groups are indicated by the allocations. This should reduce the variability of the costs and thus lead to a more stable and robust relabeling algorithm.

4.3.3.2 Extensions

Our costs (4.10) can be computed efficiently and in the examples that we have studied, they have good performance. However, it is straightforward to derive different alternatives; we can start by using robust measures of location and dispersion for the groups, e.g. the median and median absolute deviation. Thus, in (4.11) we substitute $(\overline{y}_j, s_j)$ by $(\text{med}_j, \text{mad}_j)$. With this change, the costs (4.10) will be resistant to outliers. This will be useful when the sampler visits the tails of the posterior distribution. To find $(\widehat{\mathsf{m}}_j, \widehat{\mathsf{s}}_j)$, for $j = 1, \ldots, k$, in Algorithm 4.14, we would use

$$\text{med}_j^t = \underset{\{i:z_i^t=j\}}{\text{med}}(y_i) \quad \text{and} \quad \text{mad}_j^t = \underset{\{i:z_i^t=j\}}{\text{med}}(|y_i - \text{med}_j^t|).$$

The calculation of these statistics will lead to a slower algorithm but providing accurate component specific inference. Another alternative is to use in (4.8) the costs

$$c_{l,j} = -\sum_{\{i:z_i=j\}} \log\left\{f(y_i|\theta_l)\right\},$$

thus devising label switching algorithms for particular component's distributions. The idea again will be to find posterior estimates for θ_l via (4.12): using the data and the allocations. Under this setting, for a mixture of normals, the costs will be given by

$$c_{l,j} = \frac{n_j}{2}\log(2\pi\sigma_l^2) + \frac{1}{2}\sum_{\{i:z_i=j\}}\left(\frac{y_i - \mu_l}{\sigma_l}\right)^2. \tag{4.13}$$

To estimate μ_l and σ_l we can use again (4.11). Importantly, connecting all the elements of the costs directly with the data.

4.4 Comparisons

In this section, using simulated and real data sets, we compare the performance of the relabeling algorithms: Data, ECR and KL. First, all comparisons are carried out under a univariate setting. Then we move to the multivariate case.

4.4.1 Univariate mixtures

For our first comparison, we take samples from three models involving mixtures:

$$\begin{aligned}
\text{Model 1}: &\quad 0.4\mathrm{N}(y|0.63, 0.00032) + 0.6\mathrm{N}(y|0.65, 0.00016).\\
\text{Model 2}: &\quad 0.25\mathrm{N}(y|-3, 1) + 0.25\mathrm{N}(y|-1, 1) + 0.25\mathrm{N}(y|1, 1) + 0.25\mathrm{N}(y|3, 1).\\
\text{Model 3}: &\quad 0.20\mathrm{N}(y|19, 5) + 0.20\mathrm{N}(y|19, 1) + 0.25\mathrm{N}(y|23, 1)\\
&\qquad + 0.20\mathrm{N}(y|29, 0.5) + 0.15\mathrm{N}(y|33, 2).
\end{aligned}$$

We generated samples of size: $n = 1000$, $n = 200$ and $n = 600$, from Models 1, 2 and 3, respectively. To improve comparability, the samples were not randomly drawn. We evaluated the quantile function of each mixture on a grid in $(0, 1)$, and a friendly implementation of this can be found in the package `nor1mix` of R, see Mächler (2011). The idea is that under non-informative priors, inference is as good as the sample and since the objective is to compare our proposal with other alternatives, using as reference the true values is the best way to proceed. This type of data has been used before by Nobile and Fearnside (2007) to illustrate the performance of their trans-dimensional MCMC for mixture models.

Model 1 was designed to exhibit the same characteristics as the crab data which was first analyzed by Pearson (1894), and comprises the measurements of the ratio of forehead to body length of 1000 crabs. Model 2 is a mixture of symmetric and poorly separated

components, and Model 3 has been used by Papastamoulis and Iliopoulos (2010) to demonstrate the performance of the ECR algorithm. All mixtures have overlapping components and present challenging scenarios for any relabeling algorithm.

For the real data we have chosen the acidity and enzyme data. The acidity data set concerns the log scale of the acidity index measured in a sample of 155 lakes in north-central Wisconsin. The enzyme data set concerns the distribution of enzymatic activity in the blood of an enzyme involved in the metabolism of a carcinogenic substance, among a group of 245 unrelated individuals. These data sets have been analyzed by Richardson and Green (1997).

To compare the effect on inference related to the choice of MCMC strategy we will compare posterior mean estimates, obtained via the standard Gibbs sampler, with those obtained via its reversible jump extension.

4.4.1.1 Relabeling algorithms, Gibbs sampler

For Models 1 to 3, we work with the Gibbs sampler for normal mixtures described by Richardson and Green (1997), along with their data based priors. See Section 2.2.2, for the Gibbs sampler, and Section 2.3.2.1, for the specification of the prior constants. In each case we generated an MCMC with 60,000 iterations, throwing away the fist 30,000 as a burn-in period. Then, with the output of the same MCMC, we undo the label switching using each algorithm: Data, ECR and KL. Thus, for each case, posterior mean estimates for the weights, means and variances are calculated. To measure the accuracy of the point estimates, we determined the relative error:

$$\text{Rel. Error} = \sum_{j=1}^{k} \left| \frac{\mathbb{E}(g_m(w_j, \theta_j)|\mathbf{y}) - g_m(w_j, \theta_j)}{g_m(w_j, \theta_j)} \right|,$$

where the posterior means for the functions $g_1(w_j, \theta_j) = w_j$ and $g_2(w_j, \theta_j) = \theta_j$ will be estimated via (4.5). To have a fair comparison, this was repeated 100 times.

In Table 4.2 we present the first comparison with the data set from Model 1. In this case, looking at the column of the average relative errors, it is clear that Data and KL

algorithms are more accurate than the ECR strategy. There is no difference in precision between Data and KL. In Figures 4.1 a), b) and c) we display the traces for the means (just 10,000 iterations) generated after the algorithms Data, ECR and KL, respectively, were used to undo the label switching. From these graphics, it is clear that while the Data and KL strategies introduce the constraint $\mu_1 < \mu_2$, over all 10,000 iterations, the ECR relabeling introduced the same constraint on iterations where regions of high probability of the posterior distribution were explored. Hence, as Papastamoulis and Iliopoulos (2010) proposed more alternatives to obtain good vectors of allocations, we changed $\mathbf{g^{MAP}}$ for the true single best clustering (there is no better alternative) on the ECR algorithm, and the results were then the same as with the Data and KL strategies. To test convergence problems we performed individual runs of 500,000 iterations (taking 400,000 as a burn in period), but the results calculated via the ECR algorithm and $\mathbf{g^{MAP}}$ were similar to those presented in Table 4.2 and Figure 4.1.

j		1	2	Average	Std.Dev
True	w_j	0.4	0.6	Rel. Error	Rel. Error
Data	$\overline{\mathbb{E}}(w_j\|\mathbf{y})$	0.349	0.651	0.214	0.064
ECR	$\overline{\mathbb{E}}(w_j\|\mathbf{y})$	0.332	0.668	0.283	0.050
KL	$\overline{\mathbb{E}}(w_j\|\mathbf{y})$	0.349	0.651	0.214	0.064
True	μ_j	0.63	0.65		
Data	$\overline{\mathbb{E}}(\mu_j\|\mathbf{y})$	0.627	0.649	0.006	0.001
ECR	$\overline{\mathbb{E}}(\mu_j\|\mathbf{y})$	0.628	0.648	0.006	0.001
KL	$\overline{\mathbb{E}}(\mu_j\|\mathbf{y})$	0.627	0.649	0.006	0.001
True	σ_j^2	0.00032	0.00016		
Data	$\overline{\mathbb{E}}(\sigma_j^2\|\mathbf{y})$	0.000277	0.000165	0.166	0.033
ECR	$\overline{\mathbb{E}}(\sigma_j^2\|\mathbf{y})$	0.000262	0.000180	0.308	0.035
KL	$\overline{\mathbb{E}}(\sigma_j^2\|\mathbf{y})$	0.000277	0.000165	0.166	0.033

Table 4.2: Average of posterior mean estimates and relative errors across one hundred experiments for the data set from Model 1.

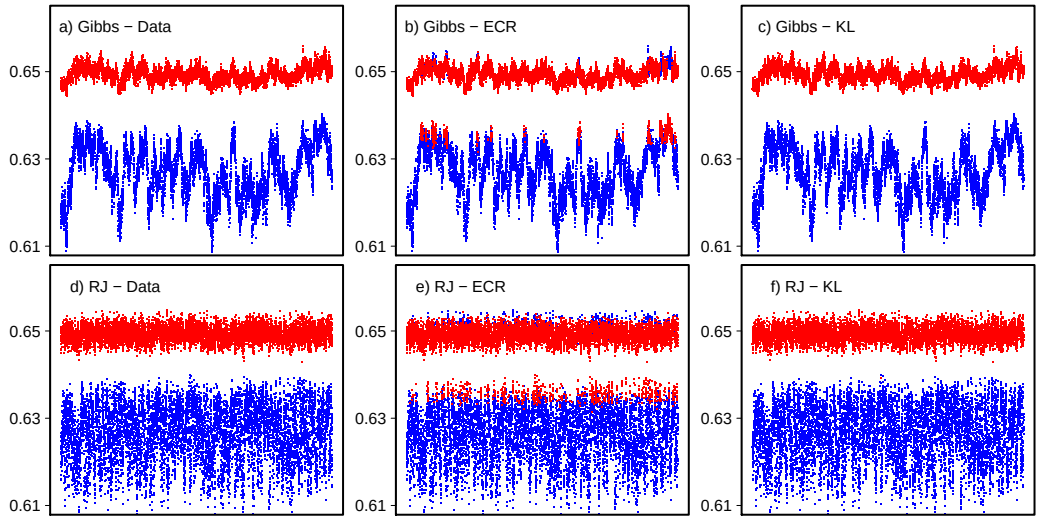

Figure 4.1: Trace for the means, data from Model 1: Gibbs sampler (upper row) and reversible jump.

For our second comparison we use the data set from Model 2, and the results are shown in Table 4.3. In this case we have a symmetric mixture of four components and where the only difference between components is the mean. From average relative errors for the weights and means, in this case, we see that Data relabeling is far more accurate than either ECR and KL strategies. Estimates of variances are equally bad across the three algorithms. If we look at the scaled components predictive and single best clustering, presented in Figure 4.2, we observe again that Data relabeling performs much better than the alternatives. True scaled densities were plotted via $w_j N(y|\mu_j, \sigma_j^2)$ and the true single best clustering was plotted via (y_i, u_i), for $i = 1, \ldots, n$ where $u_i \sim_{\text{i.i.d.}} \text{U}(\varepsilon_1, \varepsilon_2)$, then the single best clustering was used as the labels for the dots (the true single best clustering was determined using equation (4.1)). Here we can see that the single best clustering achieved with the Data relabeling, Figure 4.2 d), is almost identical to the one calculated with the true values, Figure 4.2 a). For curiosity, we used the true single best clustering instead of $\mathbf{g}^{\mathbf{MAP}}$ on the ECR algorithm and the results shown in Table 4.3 with the Data relabeling remain the more accurate. To test convergence problems,

we ran the Gibbs sampler for 1,000,000 iterations discarding the first 900,000, and the results were almost identical to those described above. Our insistence in maintaining a maximum of 100,000 iterations for the analysis lies in the storage requirements for the classification probabilities and the time needed by the KL algorithm to converge (in this case approx 16 min). See Section 4.4.3 for more about the storage requirements and Table 4.5 for a comparison of system times.

j		1	2	3	4	Average	Std.Dev
True	w_j	0.25	0.25	0.25	.25	Rel. Error	Rel. Error
Data	$\overline{\mathbb{IE}}(w_j\|\mathbf{y})$	0.224	0.277	0.273	0.226	0.442	0.089
ECR	$\overline{\mathbb{IE}}(w_j\|\mathbf{y})$	0.381	0.122	0.122	0.375	1.993	0.048
KL	$\overline{\mathbb{IE}}(w_j\|\mathbf{y})$	0.328	0.168	0.176	0.328	1.587	0.281
True	μ_j	-3	-1	1	3		
Data	$\overline{\mathbb{IE}}(\mu_j\|\mathbf{y})$	-2.191	-0.886	0.879	2.202	0.776	0.065
ECR	$\overline{\mathbb{IE}}(\mu_j\|\mathbf{y})$	-1.898	-0.480	0.453	1.930	1.801	0.507
KL	$\overline{\mathbb{IE}}(\mu_j\|\mathbf{y})$	-2.165	-0.406	0.405	2.171	2.064	0.455
True	σ_j^2	1	1	1	1		
Data	$\overline{\mathbb{IE}}(\sigma_j^2\|\mathbf{y})$	1.916	2.734	2.729	1.910	5.289	0.085
ECR	$\overline{\mathbb{IE}}(\sigma_j^2\|\mathbf{y})$	2.270	2.388	2.396	2.235	5.289	0.085
KL	$\overline{\mathbb{IE}}(\sigma_j^2\|\mathbf{y})$	2.039	2.612	2.606	2.031	5.289	0.085

Table 4.3: Average of posterior mean estimates and relative errors across one hundred experiments for the data set from Model 2.

In Model 3 we have three separated components: components three four and five. Two components have the same mean and weight: components one and two. But component one has a larger variance than component two. The results are in Table 4.4. Here, posterior mean estimates for the separated components are acceptable, but for components one and two we observe difficulties. In this case, posterior mean estimates for the weights calculated via our relabeling algorithm are more accurate. For the other variables, the ECR algorithm performs marginally better.

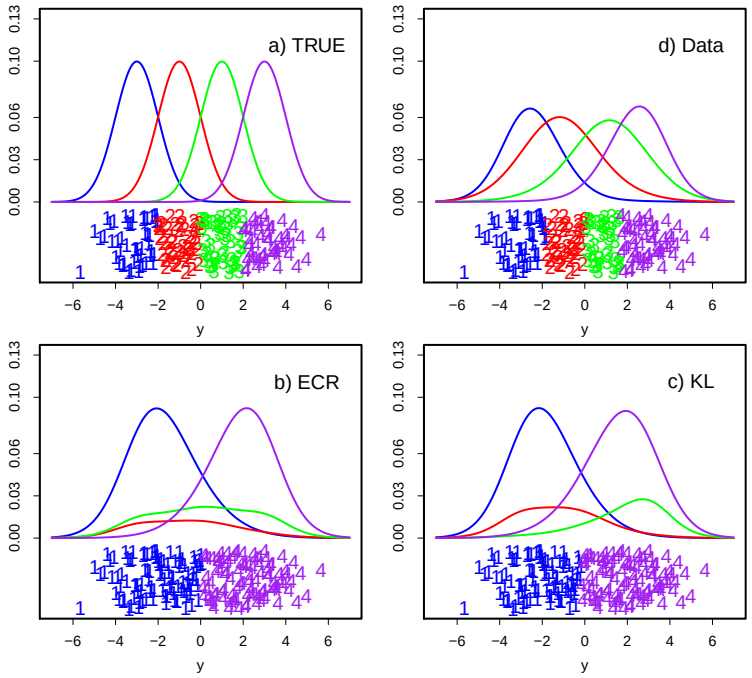

Figure 4.2: True, estimated scaled densities and single best clustering for Model 2.

j		1	2	3	4	5	Average	Std.Dev
True	w_j	0.2	0.2	0.25	0.2	0.15	Rel. Error	Rel. Error
Data	$\overline{\mathbb{IE}}(w_j\|\mathbf{y})$	0.218	0.177	0.256	0.2	0.149	0.818	0.230
ECR	$\overline{\mathbb{IE}}(w_j\|\mathbf{y})$	0.097	0.298	0.256	0.2	0.149	1.044	0.051
KL	$\overline{\mathbb{IE}}(w_j\|\mathbf{y})$	0.101	0.294	0.256	0.2	0.149	1.061	0.050
True	μ_j	19	19	23	29	33		
Data	$\overline{\mathbb{IE}}(\mu_j\|\mathbf{y})$	18.833	18.824	22.988	29.011	33.003	0.047	0.013
ECR	$\overline{\mathbb{IE}}(\mu_j\|\mathbf{y})$	18.590	19.067	22.988	29.011	33.004	0.029	0.014
KL	$\overline{\mathbb{IE}}(\mu_j\|\mathbf{y})$	18.569	19.089	22.988	29.011	33.003	0.031	0.014
True	σ_j^2	5	1	1	0.5	2		
Data	$\overline{\mathbb{IE}}(\sigma_j^2\|\mathbf{y})$	2.654	2.145	1.041	0.546	1.988	1.754	0.253
ECR	$\overline{\mathbb{IE}}(\sigma_j^2\|\mathbf{y})$	2.695	2.105	1.042	0.546	1.987	1.706	0.093
KL	$\overline{\mathbb{IE}}(\sigma_j^2\|\mathbf{y})$	2.609	2.191	1.041	0.546	1.988	1.808	0.106

Table 4.4: Average of posterior mean estimates and relative errors across one hundred experiments for the data set of Model 3.

For the acidity and enzyme data we ran the Gibbs sampler for normal mixtures described by Richardson and Green (1997), along with their data based priors. Here, assuming $k = 3$ and $k = 4$, respectively, we generated 200,000 iterations throwing away the fist 100,000 as a burn-in period. Then, with the output of the same MCMC, we undo the label switching using each algorithm: Data, ECR and KL. We display in Figure 4.3, estimated scaled densities and single best clustering. For the acidity data we see in Figure 4.3 a), b) and c) that the single best clustering produced with the three algorithms is rather similar, however there are differences in the estimated scaled densities, see group two (in red). For the enzyme data, Figure 4.3 d), e) and f), we note differences in the single best clustering produced with the three algorithms, see group four (in purple). Also, there are differences in the estimates for the scaled densities for the same component.

In Table 4.5 we present the recorded average time required to undo the label switching for each algorithm. For the Data relabeling we are differentiating between the registered time to find estimates, Algorithm 4.14, and the actual relabeling time, Algorithm 4.15. For the ECR strategy the time required for the calculation of the log posterior distribution and the time to find and extract the MAP estimate was not recorded. Given the order of the system times displayed in Table 4.5, seconds, we believe that if these calculations are considered, Data and ECR relabeling should be equally fast. We remark that this is for the univariate case.

4.4.1.2 Relabeling algorithms, reversible jump

Here, the results obtained in the last section for Models 1 to 3 are compared with estimates obtained via the reversible jump sampler, for normal mixtures, described by Richardson and Green (1997) and their data based priors. See Section 2.3.2. First, to obtain convergence, we generated 500,000 iterations and kept the last 100,000 for Model 1, and 200,000 for Models 2 and 3. Then the output relevant to a given k was extracted:

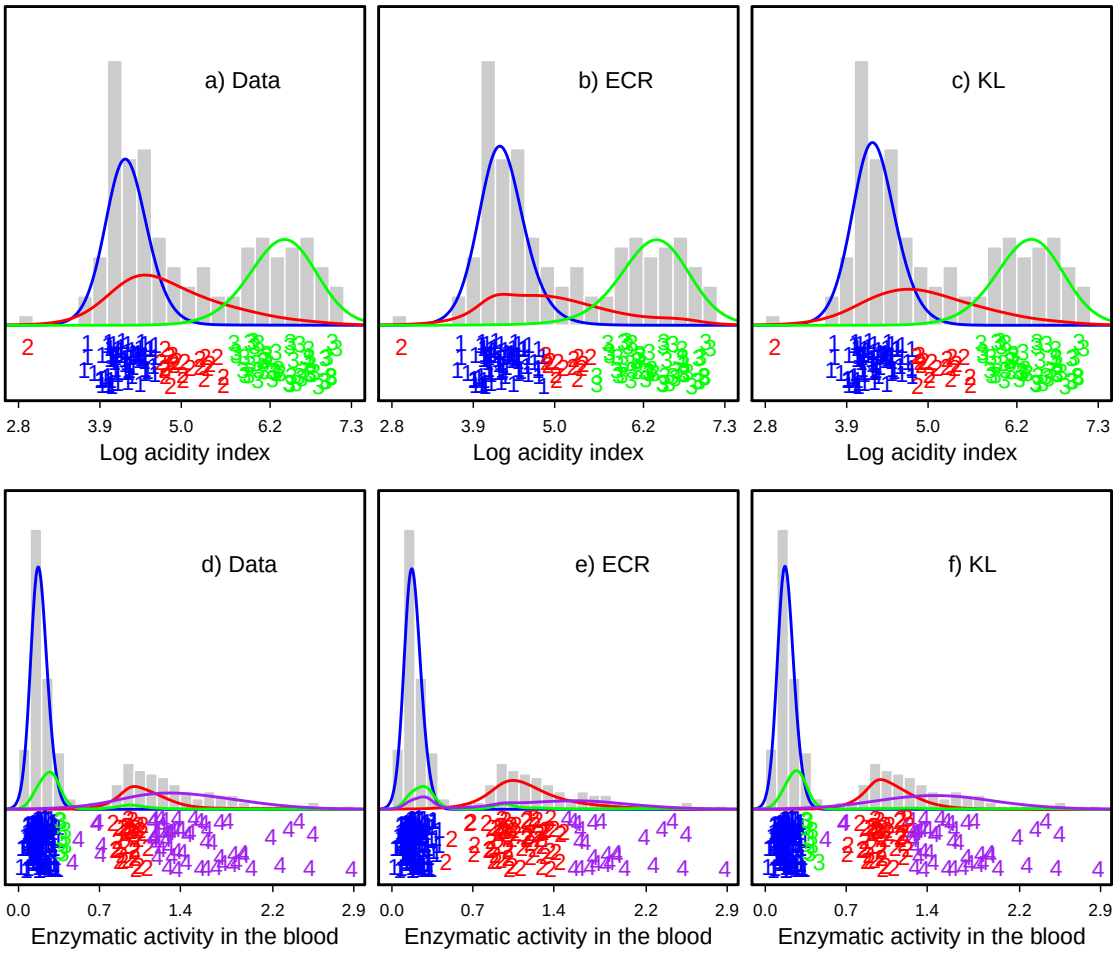

Figure 4.3: Estimated scaled densities and single best clustering for the acidity (upper row) and enzyme data.

Data Set	k	n	N	Data Find Est.	Data Relabeling	ECR	KL
Model 1	2	1000	30,000	4.855 s	4.919 s	4.853 s	1.12 min
Model 2	4	200	30,000	1.053 s	1.088 s	1.111 s	2.66 min
Model 3	5	600	30,000	3.012 s	3.020 s	2.992 s	4.45 min
Acidity	3	155	100,000	3.35 s	3.56 s	3.67 s	1.85 min
Enzyme	4	245	100,000	5.412 s	5.61 s	5.541 s	7.03 min

Table 4.5: Average system time to undo the label switching across one hundred experiments (for the acidity and enzyme data only one experiment was performed).

$k = 2$, 4 and 5 for the data sets of Model 1, 2 and 3, respectively. Finally, after relabeling the output, posterior mean estimates were calculated. In this case, because reversible jump is computationally more expensive than the Gibbs sampler, we did not carry out 100 experiments.

A remarkable difference between the two samplers is how they explore the support of the posterior distribution. The reversible jump sampler mixes better, and this can clearly be seen in Figure 4.1 where traces for the means for the data set from Model 1 are displayed, for only 10,000 iterations. Graphics d), e) and f) are from the reversible jump's output and undoing the label switching using the Data, ECR and KL algorithms, respectively. Graphics a), b) and c) are the corresponding traces generated with the output of the Gibbs sampler.

For Model 1, with reversible jump, we did not observe major improvements in accuracy and all the estimates were very close to those shown in Table 4.2.

For the data set from Model 2, we observed an improvement in the accuracy for the estimation of the weights. For example, with the Reversible Jump and Data relabeling, we achieved a relative error of 0.269, while the average relative error via the Gibbs sampler was 0.442, with a standard deviation of just 0.089. Similar improvements were obtained with the ECR and KL strategies. Curiously, estimates for the means and variances went in the opposite direction. In the case of the Data relabeling, with the Gibbs sampler, we obtained an average relative error of 0.776 and a standard deviation of 0.065, and via reversible jump a relative error of 0.923 was recorded.

For Model 3 we observed again an improvement in the estimation of the weights. This time it happened just with the the Data relabeling: from an average relative error of 0.818 (and a standard deviation of 0.230) it went to 0.455. All the other estimates were of the same order as those displayed in Table 4.4.

To conclude this section, it is important to say that the computational cost of the Reversible Jump sampler is far greater than that of the Gibbs sampler. For example,

to obtain the results for Model 1, it took almost nine minutes to run the MCMC and a further 4 minutes to extract and calculate the classification probabilities for $k = 2$. For the standard Gibbs sampler, less that 2 minutes were needed overall. Hence, given the evidence observed in our experiments, we see no reason to use the a trans-dimensional MCMC rather than the standard Gibbs sampler.

4.4.2 Multivariate normal mixtures

Here we extend the loss function (4.9) to allow for multivariate data. In this case $\mathbf{y} = \{\{y_{i,r}\}_{r=1}^{p}\}_{i=1}^{n}$, and hence each cluster has p means and p standard deviations: $\mathsf{m}_j = \{\mathsf{m}_{j,r}\}_{r=1}^{p}$ and $\mathsf{s}_j = \{\mathsf{s}_{j,r}\}_{r=1}^{p}$. With this consideration, the loss function (4.9) becomes

$$\sum_{l=1}^{k}\sum_{j=1}^{k} n_{\rho_t(j)}^{t} \sum_{\{i:\rho_t(z_i^t)=j\}} \left\{ \sum_{r=1}^{p} \left(\frac{y_{i,r} - \mathsf{m}_{l,r}}{\mathsf{s}_{l,r}} \right)^2 \right\}, \tag{4.14}$$

and thus the costs become

$$c_{l,j}^{t} = n_j^t \sum_{\{i:z_i^t=j\}} \left\{ \sum_{r=1}^{p} \left(\frac{y_{i,r} - \mathsf{m}_{l,r}}{\mathsf{s}_{l,r}} \right)^2 \right\}.$$

To illustrate the performance of our methodology in a multivariate setting, the Gibbs sampler for multivariate normals described in Stephens (1997), with default priors, was implemented. We simulated $n = 200$ observations from the distribution $\sum_{j=1}^{4} w_j \, \mathrm{N}_2(\mu_j, \Sigma_j)$ with actual values shown in Table 4.6 (note that $\Sigma = (\Sigma_{11}, \Sigma_{12} = \Sigma_{21}, \Sigma_{22})$). This mixture model has been considered by Papastamoulis and Iliopoulos (2010). In our experiments we took $k = 4$. For the analysis we kept the last 30,000 iterations from a chain of 100,000 iterations.

In Figure 4.4 a) we display the level curves calculated via the true model along with the true single best clustering. In Figures 4.4 b), c) and d) we display level curves of the plug-in density estimates and single best clustering generated via ECR, Data and KL algorithms, respectively. In this case the results obtained with each algorithm are

j	w_j	μ_j	Σ_j
1	0.25	(4.5, -2.5)	(0.5, -0.25, 0.5)
2	0.25	(-3.0, 4.0)	(0.5, -0.25, 0.5)
3	0.25	(6.5, 7.0)	(4, 2.5, 4)
4	0.25	(7.0, -3.0)	(4, 2.5, 9)

Table 4.6: Parameters for the simulated data.

identical. We repeated the experiment several times and there was no difference between the estimates generated via the three algorithms.

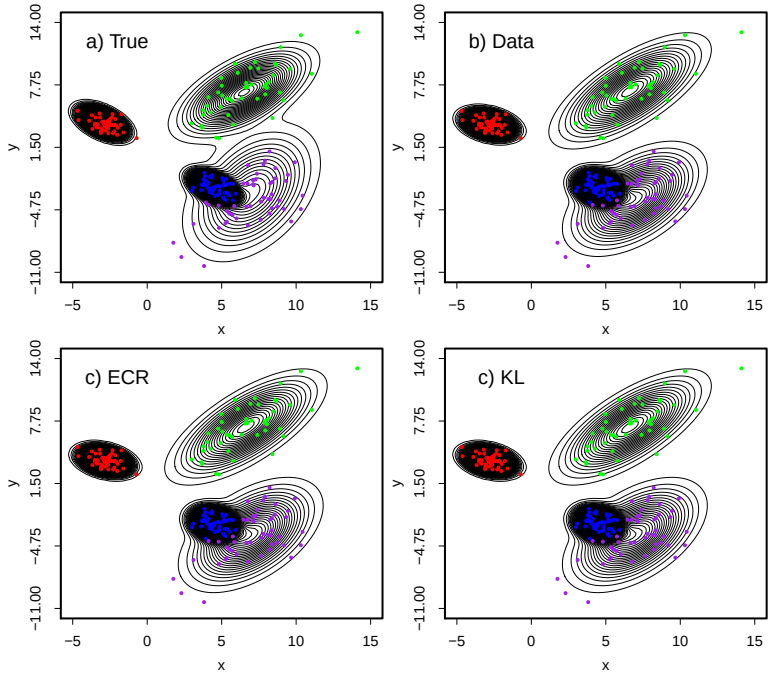

Figure 4.4: True, estimated scaled densities and single best clustering.

The time required to find estimates for the Data algorithm was 1.352 seconds, and then it took a further 1.525 seconds to undo the label switching. With the ECR and KL strategies it took 1.457 and 30.319 seconds, respectively, to undo the label switching. Again, we did not measure the time needed to find the MAP estimate for the ECR relabeling.

4.4.3 Computations and storage requirements

All the experiments were performed in a Dell Precision M4400 (processor Intel core 2 Duo at 2.26 GHz) running Linux openSUSE 12.1. We coded our approach in **C** and used the `.C` interface to R to have a friendly data input interface, see Peng and Leeuw (2002) for a good introduction. The analysis and graphics were done in R. To solve (4.8) efficiently via the Hungarian method we borrowed the function `solve_LSAP` from the library `clue` of R whose source code is written in **C**, see Hornik (2012). Thus allowing us to incorporate it into our own **C** code directly. The reported CPU times were measured using the function `system.time` of R.

One of the main criticisms of KL relabeling is its storage requirements. In our implementation, the classification probabilities were stored in a text file directly, at each iteration of the Gibbs sampler, and we worked with 16 digits of precision. Then our KL implementation loaded all the matrices of classification probabilities from this text file. With this approach the memory used to store one classification probability in a text file was 18 bytes (0. + 16 digits), then 2 bytes to print a space after each classification probability ("\t") and 1 byte to print in the next row of the text file ("\n"). Thus, the text file size to store N matrices of classification probabilities of dimension $n \times k$ was of

$$\text{TFScp}(N, n, k) = (18kn + 2kn + n)N = n(20k + 1)N \quad \text{bytes.} \tag{4.15}$$

On the other hand, the text file to store N allocation vectors of dimensions $1 \times N$ used by Data and ECR relabelings was

$$\text{TFSalloc}(N, n) = (3n + 1)N \quad \text{bytes.} \tag{4.16}$$

This follows since 1 byte is needed to store one allocation, 2 bytes to print a space after each allocation, and 1 byte to print in the next row of the text file for each allocation vector.

Then, with the precision considered, we see that the text file size to store the classification probabilities is approximately $7k$ times larger than the one used to store the

vectors of allocations. We can use (4.15) and (4.16) to work out the size of the text files used in our experiments. These are displayed in Table 4.7. Note that 1 MB = 1024^2 bytes and 1 GB = 1024^3 bytes.

Data Set	k	n	N	TFScp	TFSalloc
Model 1	2	1000	30,000	1.1 GB	85.9 MB
Model 2	4	200	30,000	463.5 MB	17.2 MB
Model 3	5	600	30,000	1.7 GB	51.5 MB

Table 4.7: Text file size for the classification probabilities and latent allocations.

The size of the text files shown in Table 4.7 are easily handled by modern day computers, but to test our every day laptop we increased N gradually running the algorithms with the data set from Model 3. As our laptop's processor is of 32 bytes, we experienced problems to store text files larger than 2 GB, however this was easily solved enabling the large file support in Linux, see Jaeger (2005). When $N = 110,000$ was reached, and hence 6.2 GB of size for the classification probabilities text file, the KL algorithm crashed: when loading all the matrices of classification probabilities. If the aim when performing component specific inference is classification analysis, then the N matrices of classification probabilities sampled throughout the MCMC are needed. If the Data relabeling is used, we can store all the classification probabilities directly into a text file, as described before, and then calculate (4.3) reading just one matrix of classification probabilities at a time. In our experiments text file sizes of classification probabilities of 15 GB of memory were handled without any problem. A possible solution for the KL strategy to handle larger text files could be to read one matrix of classification probabilities at a time and return the file pointer to the beginning of the text file, at the end of each iteration. However, this would slow the KL algorithm even more.

4.5 Discussion

4.5.1 Conclusions

We have described the key ideas for a deterministic relabeling algorithm and from it have derived an easy and efficient solution to the label switching problem. Our proposed solution, the Data relabeling, lies in the meaning of the relationship between the allocation variables (2.2) and the observations. If the sampler has converged, observations being allocated together should remain roughly the same throughout iterations of an MCMC sampler. Hence, using a k-means type of diverging measure, (4.9), we are able to keep track of each cluster.

The relabeling used to make component specific inference is completely determined by the loss function. Thus, this is the most important part of a relabeling algorithm. We have incorporated the observations directly within the loss function, while alternative algorithms only use this information indirectly via the MCMC output, which has been estimated from the observations. This direct connection to the data is an appealing characteristic of our idea. As a consequence we have obtained accurate posterior mean estimates, acceptable scaled predictive densities and good single best clusterings.

To implement the Data relabeling algorithm all that is needed are the observations and the latent allocations. There is no need to embed additional calculations within the MCMC sampler, as with the ECR strategy. With this, we first find estimates for the means and standard deviations of each cluster and then a relabeling that recovers identifiability for the mixture components. We stress that care is needed when finding estimates. It could appear that a fast solution is the MAP estimate. However, in some cases it fails to completely isolate one of the symmetric modes of the posterior distribution. This was seen in our experiments when the ECR was used to analyse a simulated version of the Crab data.

Our strategy does not remain the same if the dimension of the data changes. However,

the modification needed to extend the algorithm to a multivariate framework is a simple loop, see (4.9) and (4.14).

A lack of label switching in the standard Gibbs sampler has been perceived as a convergence problem. However, in our comparison of estimates among samplers, we saw no evidence to support this idea. At most, in some cases, we observed that estimates for the weights were more accurate when the reversible jump sampler was used. Nevertheless, this improvement was not preserved for the remaining parameters and hence we see little motivation to search for samplers more complicated than the standard Gibbs sampler.

4.5.2 Future Work

Our relabeling algorithm assumes there is no genuine multimodality (as KL and ECR strategies do). Recent work done by Chopin, Lelièvre, and Stoltz (2012) proposes a method to enhance the sampling of MCMC strategies, and efficiently exploring symmetric and genuine modes of a univariate normal mixture model (this is a clear improvement over previous non-trivial samplers designed with that porpoise). Therefore, it would be of interest to compare the performance of the three algorithms under this context, and determine if the use of such sampler improves the accuracy of the estimates or not. Further, we noted that the classic example where genuine multimodality appears is when fitting the galaxy data with a mixture of three Student's t-distributions, t_4, (see Stephens (1997) p. 59 and Papastamoulis and Iliopoulos (2010)). However, when the galaxy data is fitted with a mixture of three normal distributions, then there is no strong evidence of genuine multimodality. Hence, in this case, the presence of genuine multimodality is model dependent, and it would be important to investigate if (in general) genuine multimodality is a symptom of model inadequacy or at least the causes for its appearance.

Label switching can be considered under a general missing data model framework that includes as special cases finite mixtures, hidden Markov models, and Markov random fields, see Papastamoulis and Iliopoulos (2011). In this thesis we restrict our attention to

finite mixtures, however, the aim is to extend our ideas to the general case. An immediate application under the general setting is the mixture of Dirichlet process (MDP) model, as discussed in Chapter 3. In this setting, to deal with an infinite measure, there are two alternative ideas: marginal and conditional methods. If the Dirichlet process is integrated out via the so called marginal algorithms, see Section 3.2.1, the labels of the clusters become unidentifiable, see Papaspiliopoulos and Roberts (2008). For the conditional algorithms, see Section 3.2.2, the weights of the Dirichlet process are only weakly identifiable, thus there is label switching. Note that to improve the mixing properties of the sampler label switching moves are introduced to the MCMC algorithm, see Section 3.2.2.4. In both cases, to recover identifiability we could proceed as with reversible jump, extracting the MCMC output relevant to a given k and using the Data relabeling.

Chapter 5

Nonparametric Mixture Modeling with Unimodal Kernels

As discussed in Chapter 1 a common theme in mixture models is the use of the normal distribution as the "benchmark" components distribution. However, if a cluster is skewed or heavy tailed, then the normal distribution will be inefficient and many may be needed to model a single cluster. In this Chapter, we present an attempt to solve this problem. The aim is to ensure the number of clusters within the data coincides with the estimate of the number of components. With this objective, first, via the MDP model we introduce a new family of nonparametric unimodal distributions, which has large support over the space of unimodal distributions. Then, we use this unimodal distribution as the components distribution in a finite mixture model, where each part of the model has a proper meaning, and k is modeled explicitly. Hence, given k, the model is a finite mixture of k unimodal densities, each of which is modeled nonparametrically.

Much effort needs to be dedicated to the MCMC sampler. In fact, we derive a hybrid MCMC strategy, see Section A.4.4. There are three key points here. First, the introduction of some latent allocation variables (as with every mixture model). Second, the slice sampling ideas of Kalli, Griffin, and Walker (2011) to truncate the number of

variables to a finite number, see Section 3.2.2. Third, following Godsill (2001), a trans-dimensional step is devised, for an entire stochastic process, writing a joint distribution on the product space of candidate models and performing a Metropolis-Hastings in the usual way, see Section 2.3. To make inference for individual components and cluster analysis the well known problem of label switching must be addressed. We use our Data relabeling algorithm, to "undo" the label switching and recover identifiability of the mixture parameters, see Section 4.3.

Hence, this Chapter brings together the three previous Chapters. We will work under a finite mixture modelling setting, Chapter 2. The MDP model will be used for its original purpose, i.e. to define a flexible random density, Chapter 3, and to deal with the label switching problem we will use our relabeling algorithm, Chapter 4.

5.1 The model

The model for the data will be a finite mixture model with k components, and k is assumed unknown, written as:

$$f_{\mathbf{G}}(y|\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}, k) = \sum_{j=1}^{k} w_j \, f_{G_j}(y|\mu_j, \lambda_j), \tag{5.1}$$

with $\mathbf{w} = \{w_j\}_{j=1}^k$, $\boldsymbol{\lambda} = \{\lambda_j\}_{j=1}^k, \boldsymbol{\mu} = \{\mu_j\}_{j=1}^k$ and $\mathbf{G} = \{G_j\}_{j=1}^k$. The weights, $\mathbf{w}$, are non-negative and sum to one; and for each j, $f_{G_j}(y|\mu_j, \lambda_j)$ is a univariate unimodal distribution which can be fully characterized by a distribution function G_j, and also with the parameter λ_j, which determines the asymmetry of f_{G_j}, and μ_j, which determines the location of f_{G_j}. All the other aspects of f_{G_j} can be determined by the moments of the distribution G_j.

To begin, we describe the family of univariate unimodal density functions which will serve as the components of the mixture model (5.1).

5.1.1 Unimodal kernels

We start with the MDP model, Lo (1984), and chose a Dirichlet process prior, $\mathrm{DP}(c, G_0)$, over G as described in Section 3.2:

$$f_G(y) = \int_\Theta k(y|\theta) G(\mathbf{d}\theta) = \sum_{s=1}^{\infty} w_s k(y|\theta_s). \tag{5.2}$$

If the kernel $k(y|\mu, \sigma^2) = \mathrm{N}(y|\mu, \sigma^2)$ is taken, then (5.2) defines a multi-modal distribution, which is not what we want. On the other hand, if

$$k(y|\theta, \mu) = \mathrm{U}(y|\mu - \theta, \mu + \theta), \quad \text{with} \quad \theta \in \mathbb{R}^+,$$

then it is well known that scale mixtures of uniforms coincide with the class of unimodal and symmetric distributions, p. 158 of Feller (1971):

$$\begin{aligned} f_G(y|\mu) &= \int_0^\infty \mathrm{U}(y|\mu - \theta, \mu + \theta) G(\mathbf{d}\theta), \\ &= \sum_{s=1}^{\infty} w_s \, \mathrm{U}(y|\mu - \theta_s, \mu + \theta_s), \end{aligned} \tag{5.3}$$

where $\mathrm{U}(y|a, b)$ is the uniform density on (a, b).

In this case $f_G(y|\mu)$ is a symmetric random density with mode μ, and where its variance and kurtosis are determined by G.

Brunner and Lo (1989) and Quintana, Steel, and Ferreira (2009), among others, have worked with (5.3). But in our case, we want to extend (5.3) to include asymmetry. A possible option is to use the proposal of Kottas and Gelfand (2001) where they model asymmetry using two independent Dirichlet processes G_1 and G_2, i.e.

$$f_{G_1,G_2}(y|\mu) = \frac{1}{2}\left\{ \int \frac{1}{\theta} \mathbb{1}_{(\mu-\theta,\mu)}^{(y)} G_1(\mathbf{d}\theta) + \int \frac{1}{\theta} \mathbb{1}_{[\mu,\mu+\theta)}^{(y)} G_2(\mathbf{d}\theta) \right\}.$$

In this case, the tails of the distribution are being modeled independently of each other, which could lead to a large discontinuity at the mode. Also, if it is close to symmetric, then the model is inefficient. Instead, following the ideas of Fernandez and Steel (1998),

we can return to (5.3) and incorporate an asymmetry parameter $\lambda \in \mathbb{R}$ via

$$\begin{aligned} f_G(y|\lambda,\mu) &= \int_{\mathbb{R}^+} \mathrm{U}\left(y|\mu-\theta e^{-\lambda},\mu+\theta e^{\lambda}\right) G(\mathbf{d}\theta), \\ &= \sum_{s=1}^{\infty} w_s \,\mathrm{U}\left(y|\mu-\theta_s e^{-\lambda},\mu+\theta_s e^{\lambda}\right). \end{aligned} \tag{5.4}$$

Then (5.4) defines a random unimodal density determined by (λ,μ,G); μ is the location parameter; λ the asymmetry parameter and G a distribution function. Characteristics such as variance, kurtosis, tails and higher moments are determined by G.

We do not claim that the support of (5.4) includes all the unimodal densities on the real line. But it certainly covers all symmetric unimodal distributions and a large class, sufficiently large in our estimation, of asymmetric distributions.

Then we will use (5.4) as the components distribution for the mixture model (5.1).

5.1.1.1 Prior predictive

It is important to understand what kind of shapes the unimodal distribution, (5.4), can achieve and how the parameters λ and μ and the moments of G influence it. With this aim, setting $G_0(\theta|\alpha,\beta)$ as a gamma distribution, written as $\mathrm{Ga}(\theta|\alpha,\beta)$, (parametrized such that $\mathbb{E}(\theta) = \alpha/\beta$), and denoting $\mathscr{P}$ as the Dirichlet process measure with base distribution G_0 and concentration parameter $c > 0$, we know that $G \sim \mathscr{P}$ (from Section 3.1.2). Then the prior predictive or prior guess can be calculated by integrating out G from (5.4) to yield

$$\begin{aligned} \mathbb{E}\left(f_G(y|\lambda,\mu)\right) &= \int_{\Omega} f_G(y|\lambda,\mu)\mathscr{P}(\mathbf{d}G), \\ &= \int_{\Omega}\left[\int_{\mathbb{R}^+} k(y|\theta,\lambda,\mu)G(\mathbf{d}\theta)\right]\mathscr{P}(\mathbf{d}G), \\ &= \int_{\mathbb{R}^+} k(y|\theta,\lambda,\mu)\left[\int_{\Omega} G(\mathbf{d}\theta)\mathscr{P}(\mathbf{d}G)\right], \\ &= \frac{\operatorname{sech}(\lambda)}{2}\int_{\mathbb{R}^+}\frac{1}{\theta}\,\mathbb{1}_{\left(\mu-\theta e^{-\lambda}}^{(y)},_{\mu+\theta e^{\lambda}\right)} G_0(\mathbf{d}\theta|\alpha,\beta), \\ &= \frac{\beta\operatorname{sech}(\lambda)}{2(\alpha-1)}\left(1-G_0(a(y)|\alpha-1,\beta)\right), \end{aligned} \tag{5.5}$$

with $a(y) = \max\left\{(\mu-y)e^{\lambda},(y-\mu)e^{-\lambda}\right\}$.

Note that with the choice of G_0 for $k(y|\theta, \lambda, \mu)$ to be a valid kernel it is required that $\alpha > 1$. For (5.5) to be a differentiable (smooth) function, $\alpha > 2$ is needed, and if $1 < \alpha \leq 2$ we will have a continuous function that is not differentiable at $y = \mu$. See Appendix 5.A for details.

Hence, (5.5) is a four parameter density. But the important point here is to understand how λ, μ, α and β influence the prior predictive (5.5) and ultimately (5.4). To clarify this point some graphics have been displayed in Figure 5.1. To have a measure of comparison, a plot of the standard normal distribution has been included.

We have that μ deals with the location; λ the skewness and α and β the variance and kurtosis. This can be seen for α in graphics: a), b) and d) and for λ in graphics: c), d), e) and f). From these graphics is clear that α determines the degree of kurtosis. Note in graphs e) and f) how large values of lambda ($|\lambda| > 1$) can also impact the variance. That β influences variance can be seen from graphs b) and c). Finally μ only influences the location; see graphs d) and f).

The best approximation to the normal distribution with the prior predictive is shown in graphic c), a zoom to the right tails, between 3 and 6, shows that the tails of the prior guess are slightly heavier than those of the standard normal distribution.

On studying the prior predictive (5.5) we see that contrary to the realizations of $f_G(y|\lambda, \mu)$ it is a continuous density. We observed how the random distribution G along with the parameters λ and μ influence the prior guess. This information is important because one of the aims is to approximate the posterior predictive $\mathbb{E}(f_G(y|\lambda, \mu)|\mathbf{y})$ once a sample from a population $\mathbf{y} = \{y_i\}_{i=1}^n$ has been observed. Thus the study of the prior predictive gives us an insight on the representations or shapes that the posterior predictive can achieve and how (λ, μ, G) (given the observations) can influence it. Further, since we will be working with a new distribution, this knowledge can be helpful to set the values of the unspecified constants for the priors of the model.

Later on in this Chapter we will evaluate the performance of model (5.4) by fitting

data sets from different unimodal distributions.

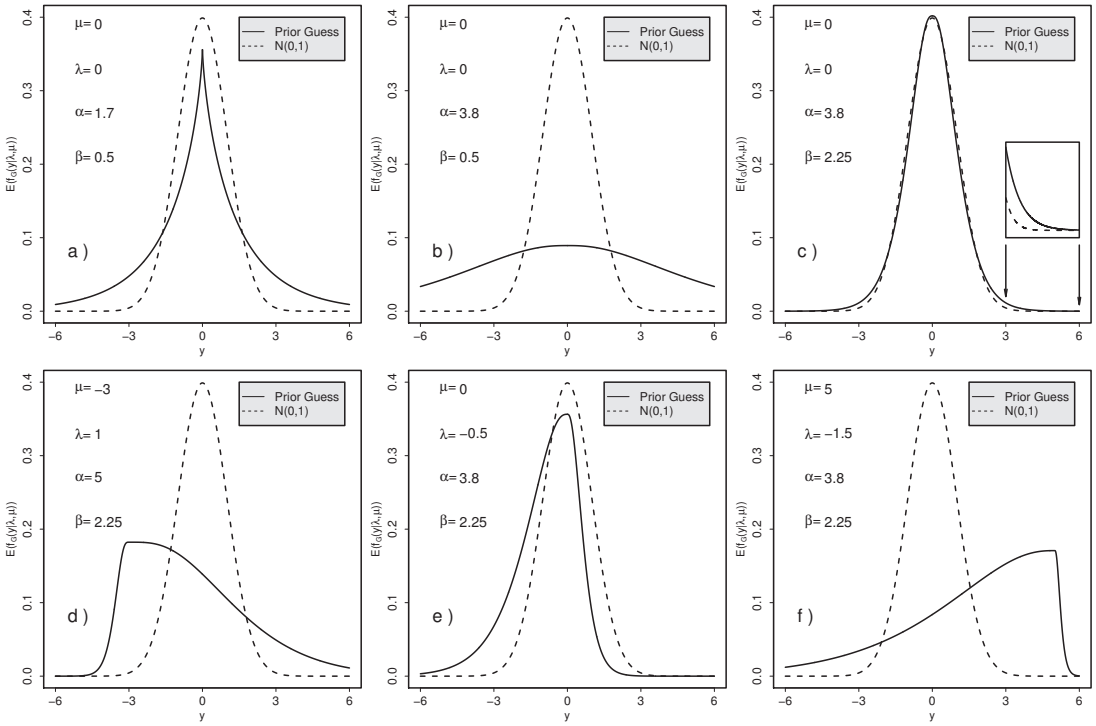

Figure 5.1: Influence of λ, μ, α and β on the prior guess.

5.1.2 Mixtures of unimodal kernels

We now write the model (5.1) as

$$f_{\mathbf{G}}(y|\mathbf{w}, \boldsymbol{\lambda}, \boldsymbol{\mu}, k) = \sum_{j=1}^{k} w_j \sum_{s=1}^{\infty} w_{js} \, \mathrm{U}\left(y|\mu_j - \theta_{js} e^{-\lambda_j}, \mu_j + \theta_{js} e^{\lambda_j}\right). \tag{5.6}$$

In (5.6) there are no assumptions about the shape of the components, the only assumption is that of unimodality. Therefore, k means something explicit here: the number of clusters modeled by a unimodal density.

5.1.2.1 Allocation variables

For the mixture of unimodal distributions we have two types of weights and therefore we will need two sets of allocation variables. Thus a joint (z_i, d_i) of latent allocation variables needs to be defined. Then $(z_i = j, d_i = s)$ will indicate that the observation y_i

has been drawn from the component j of the finite mixture ($j \in \{1, \ldots, k\}$) and from the s component of the infinite mixture for the unimodal distribution. Note that, *a priori*, (z_i, d_i) are drawn independently with distributions $p(z_i = j, d_i = s) = w_j w_{js}$, for $j = 1, \ldots, k$ and $s = 1, 2, \ldots$. Hence, given the values of (z_i, d_i), the observations are sampled from their respective components;

$$f_{\mathbf{G}}(y_i | z_i, d_i, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathrm{U}\left(y_i | \mu_{z_i} - \theta_{z_i d_i} e^{-\lambda_{z_i}}, \mu_{z_i} + \theta_{z_i d_i} e^{\lambda_{z_i}}\right).$$

Summing out the (z_i, d_i) we get back to (5.6).

These are exactly the same ideas outlined in Section 2.1 and Section 3.2.2, for the finite and infinite mixtures respectively. To sample from the full conditional of the allocation variables it will be necessary to work with

$$p(z_i = j, d_i = s | \cdots) \propto w_j w_{js}\, \mathrm{U}\left(y | \mu_j - \theta_{js} e^{-\lambda_j}, \mu_j + \theta_{js} e^{\lambda_j}\right), \tag{5.7}$$

for $j = 1, \ldots, k$ and $s = 1, 2, \ldots$. This cannot be sampled directly due to the infinite choice of s, and we will use the ideas described by Kalli, Griffin, and Walker (2011) to deal with this problem, see Section 3.2.2.

5.2 MCMC: hybrid strategy

Here we describe how to sample from the posterior distribution of model (5.6). First, in Section 5.2.1, the case for a known number of components is outlined and then the case for an unknown number of components (i.e. $k \in \{1, 2, \ldots\}$) is described in Section 2.3.

5.2.1 Known number of components

As described in Chapter 2, to tackle the problem for an unknown number of components with model (5.6) is difficult. It is better to describe how to sample the model with k fixed and then we can proceed to add the extension of a moving k.

5.2.1.1 Slice sampler for infinite mixtures

We will use slice sampler ideas to sample from (5.7), see Section 3.2.2. The idea is to add for every d_i a slice variable u_i such that

$$p(z_i, d_i, u_i|\cdots) \propto w_{z_i} w_{z_i d_i} \,\mathrm{U}(u_i|0, \xi_{d_i})\,\mathrm{U}\left(y_i|\mu_{z_i} - \theta_{z_i d_i} e^{-\lambda_{z_i}}, \mu_{z_i} + \theta_{z_i d_i} e^{\lambda_{z_i}}\right) \tag{5.8}$$

with ξ_l any positive and decreasing function in l. It is clear that integrating out the (u_i) we get back to the original distribution. The purpose of the (u_i) is to force each d_i to be from a finite set. This can be seen for example by setting $\xi_d = e^{-d}$, which is the form that we use in this Chapter. Then, from (5.8),

$$u_i \sim \mathrm{U}(u_i|0, e^{-d_i}) \quad \Rightarrow \quad u_i < e^{-d_i} \quad \Leftrightarrow \quad d_i < -\log(u_i),$$

so if $N_i = \lfloor -\log(u_i) \rfloor$ (where $\lfloor a \rfloor$ is the closest integer to a less than or equal to a) it follows that for $i = 1, 2, \ldots, n$

$$d_i \leq N_i \leq -\log(u_i) \;\Rightarrow d_i \;\in\; \{1, 2, \ldots, N_i\}\,.$$

Hence, to sample from $p(z_i, d_i, u_i|\cdots)$ we have the Gibbs sampler

$$\begin{aligned} u_i &\sim p(u_i|d_i), \\ (z_i = j, d_i = s) &\sim p(z_i = j, d_i = s|u_i, \cdots), \end{aligned}$$

with the (u_i) as uniform, i.e. $p(u_i|d_i) = \mathrm{U}\left(u_i|0, e^{-d_i}\right)$, and the allocations as

$$p(z_i = j, d_i = s|u_i, \cdots) \propto w_j w_{js} e^s \,\mathrm{U}\left(y_i|\mu_j - \theta_{js} e^{-\lambda_j}, \mu_j + \theta_{js} e^{\lambda_j}\right)$$

where $j \;\in\; \{1, \ldots, k\}$ and $s \;\in\; \{1, \ldots, N_i\}$.

If we define

$$N = \max_{i=1,\ldots,n} \{N_i\} \;\Rightarrow\; \forall \;\; i, \;\; d_i \;\in\; \{1, \ldots, N\}. \tag{5.9}$$

The variables that depend on (z_i, d_i) will be matrices: $\{\{w_{js}\}_{s=1}^{N}\}_{j=1}^{k}$ and $\{\{\theta_{js}\}_{s=1}^{N}\}_{j=1}^{k}$.

With the inclusion of these latent variables the full posterior distribution will be proportional to

$$\prod_{i=1}^{n} w_{z_i} w_{z_i d_i} \, \mathrm{U}\left(u_i | 0, e^{-d_i}\right) \mathrm{U}\left(y_i | \mu_{z_i} - \theta_{z_i d_i} e^{-\lambda_{z_i}}, \mu_{z_i} + \theta_{z_i d_i} e^{\lambda}_{z_i}\right).$$

5.2.1.2 Model priors

The prior on the weights of the finite mixture will be a Dirichlet distribution, $p(\mathbf{w}|\delta, k) = \mathrm{Dir}(\mathbf{w}|\delta, \ldots, \delta)$. For the location parameters

$$p(\boldsymbol{\mu}|\mu_0, \sigma_0^2, k) \propto k! \prod_{j=1}^{k} N(\mu_j|\mu_0, \sigma_0^2) \, \mathbb{1}\{\mu_1 < \ldots < \mu_k\} \tag{5.10}$$

the order statistics of k normal distributions.

Note that (5.10) is not to provide an identifiability constraint and break the symmetry of the likelihood of (5.6). The purpose of this prior is to impose the order needed on the location parameters to construct the invertible transformation as in Richardson and Green (1997), see Section 2.2.2.

The prior for the skewness parameters are independent and uniform: $\mathrm{U}(\lambda_j| -\epsilon, \epsilon)$ for $j = 1, \ldots, k$. To model the asymmetry of each cluster in a flexible way we included a hierarchical prior for ϵ, $p(\epsilon) = \mathrm{U}(\epsilon|0, \rho)$ for some $\rho > 0$.

In the case of the weights of the infinite mixture we will use the stick-breaking construction, thus

$$p(v_{js}) = \mathrm{Be}(v_{js}|1, c) \text{ independent for all } j \text{ and } s.$$

We centered the Dirichlet process on gamma distributions, so

$$p(\theta_{js}) = \mathrm{Ga}(\theta_{js}|\alpha, \beta_j) \text{ independent for all } j \text{ and } s.$$

In order not be too restrictive with the variance of each unimodal component a hierarchical prior for each β_j was included; $p(\beta_j) = \mathrm{Ga}(\beta_j|a, b)$.

Hence, the unspecified constants of the priors are

$$\delta, \ \mu_0, \ \sigma_0^2, \ \rho, \ c, \ \alpha, \ a \text{ and } b.$$

The smoothing parameter c, from the stick-breaking representation of the Dirichlet process, influences the size of the truncation point N, of the infinite mixture, see remark 3.10 in Chapter 3. As a result, for small values of c small values of N are obtained, thus leading to less smooth unimodal distributions. On the other hand, larger values of c support larger values for N, thus producing smoother distributions. Initially, for the fixed k case, we followed Escobar and West (1995) and imposed a gamma prior over c but the results were very similar to those where c was fixed at the mean of the gamma distribution. So, for simplicity, we omitted this hierarchical level and set c directly.

5.2.1.3 Full conditionals for a fixed k

Here we provide a careful derivation of the full conditional distributions used to construct the Gibbs sampler for the posterior distribution of the unimodal mixture, when k is known. To avoid repetition, we will define the following sets and variables:

$$\begin{aligned} A_j &= \{i : z_i = j\} \Rightarrow n_j = \#A_j \text{ and} \\ A_{js} &= \{i : z_i = j,\ d_i = s\} \Rightarrow n_{js} = \#A_{js} \end{aligned}$$

where $\#A$ is the cardinality of the set A.

- The full conditional of the weights of the finite mixture are as in Algorithm 2.1.
- The full conditionals for $\boldsymbol{\lambda} = \{\lambda_j\}_{j=1}^{k}$ are given by

$$p(\lambda_j|\cdots) \propto \mathbb{1}_{(-\epsilon,\epsilon)}^{(\lambda_j)} \operatorname{sech}(\lambda_j)^{n_j} \prod_{\{i:z_i=j\}} \mathbb{1}_{\left(-\theta_{js}e^{-\lambda_j}+\mu_j,\theta_{js}e^{\lambda_j}+\mu_j\right)}^{(y_i)},$$

so if $A_j \neq \emptyset$

$$\begin{aligned} 1 &= \prod_{\{i:z_i=j\}} \mathbb{1}_{\left(-\theta_{jd_i}e^{-\lambda_j}+\mu_j,\theta_{jd_i}e^{\lambda_j}+\mu_j\right)}^{(y_i)}, \\ &\Leftrightarrow \ \forall i \in A_j \Rightarrow -e^{-\lambda_j} < \frac{y_i-\mu_j}{\theta_{jd_i}} < e^{\lambda_j}, \\ &\Leftrightarrow \ -\min_{i\in A_j}\left\{\frac{y_i-\mu_j}{\theta_{jd_i}}\right\} < e^{-\lambda_j} \text{ and } \max_{i\in A_j}\left\{\frac{y_i-\mu_j}{\theta_{jd_i}}\right\} < e^{\lambda_j}. \end{aligned}$$

But $|\lambda_j| < \epsilon \;\Rightarrow\; e^{-\epsilon} < e^{-\lambda_j} < e^{\epsilon}$ and $e^{-\epsilon} < e^{\lambda_j} < e^{\epsilon}$ then we have found bounds $L_j < \lambda_j < U_j$ where

$$\begin{aligned} L_j &= \log\left(\max\left\{\max_{i\in A_j}\left\{\frac{y_i-\mu_j}{\theta_{jd_i}}\right\}, e^{-\epsilon}\right\}\right), \\ U_j &= -\log\left(\max\left\{-\min_{i\in A_j}\left\{\frac{y_i-\mu_j}{\theta_{jd_i}}\right\}, e^{-\epsilon}\right\}\right). \end{aligned}$$

Hence, we can write

$$p(\lambda_j|\cdots) \propto \begin{cases} \operatorname{sech}(\lambda_j)^{n_j}\, \mathbb{1}_{(L_j,U_j)}^{(\lambda_j)} & \text{if } A_j \neq \emptyset, \\ \mathrm{U}(\lambda_j|-\epsilon,\epsilon) & \text{if } A_j = \emptyset. \end{cases} \tag{5.11}$$

- The full conditionals for $\boldsymbol{\mu} = \{\mu_j\}_{j=1}^k$ are given by

$$p(\mu_j|\cdots) \propto \mathrm{N}(\mu_j|\mu_0,\sigma_0^2)\mathbb{1}_{(\mu_{j-1},\mu_{j+1})}^{(\mu_j)} \prod_{\{i:z_i=j\}} \mathbb{1}_{\left(-\theta_{jd_i}e^{-\lambda_j}+\mu_j,\theta_{jd_i}e^{\lambda_j}+\mu_j\right)}^{(y_i)},$$

and if $A_j \neq \emptyset$

$$\begin{aligned} 1 &= \prod_{\{i:z_i=j\}} \mathbb{1}_{\left(-\theta_{jd_i}e^{-\lambda_j}+\mu_j,\theta_{jd_i}e^{\lambda_j}+\mu_j\right)}^{(y_i)}, \\ &\Leftrightarrow \max_{i\in A_j}\{-\theta_{jd_i}e^{\lambda_j}+y_i\} < \mu_j < \min_{i\in A_j}\{\theta_{jd_i}e^{-\lambda_j}+y_i\}. \end{aligned}$$

There is an additional constraint; $\mu_{j-1} < \mu_j < \mu_{j+1}$, thus letting

$$\begin{aligned} I_j &= \max\{\max_{i\in A_j}\{-\theta_{jd_i}e^{\lambda_j}+y_i\}, \mu_{j-1}\}, \\ S_j &= \min\{\min_{\in A_j}\{\theta_{jd_i}e^{-\lambda_j}+y_i\}, \mu_{j+1}\}, \end{aligned}$$

the full joint for μ_j can be written as

$$p(\mu_j|\cdots) \propto \begin{cases} \mathrm{N}(\mu_j|\mu_0,\sigma_0^2)\mathbb{1}_{(I_j,S_j)}^{(\mu_j)} & \text{if } A_j \neq \emptyset \\ \mathrm{N}(\mu_j|\mu_0,\sigma_0^2)\mathbb{1}_{(\mu_{j-1},\mu_{j+1})}^{(\mu_j)} & \text{if } A_j = \emptyset, \end{cases} \tag{5.12}$$

- The infinite weights $\{\{w_{js}\}_{j=1}^k\}_{s=1}^N$ are updated as in Algorithm 3.5, with obvious changes:

$$p(v_{js}|\cdots) = \mathrm{Be}\left(v_{js}|1+n_{js}, n_j - \sum_{l=1}^{s} n_{jl} + c\right).$$

- The full conditionals for $\{\{\theta_{js}\}_{j=1}^{k}\}_{s=1}^{N}$ are derived first noting

$$p(\theta_{js}|\cdots) \propto \mathrm{Ga}(\theta_{js}|\alpha,\beta)\left(\frac{1}{\theta_{js}}\right)^{n_{js}} \prod_{\{i:z_i=j,d_i=s\}} \mathbb{1}_{\left(-\theta_{js}e^{-\lambda_j}+\mu_j,\theta_{js}e^{\lambda_j}+\mu_j\right)}^{(y_i)}.$$

If $A_{js} \neq \emptyset$

$$\begin{aligned} 1 &= \prod_{\{i:z_i=j,d_i=s\}} \mathbb{1}_{\left(-\theta_{js}e^{-\lambda_j}+\mu_j,\theta_{js}e^{\lambda_j}+\mu_j\right)}^{(y_i)} \\ &\Leftrightarrow \; -\theta_{js}e^{-\lambda_j}+\mu_j < \min_{i\in A_{js}}\{y_i\} \;\text{ and }\; \max_{i\in A_{js}}\{y_i\} < \theta_{js}e^{\lambda_j}+\mu_j \\ &\Leftrightarrow \; e^{\lambda_j}(\mu_j - \min_{i\in A_{js}}\{y_i\}) < \theta_{js} \;\text{ and }\; e^{-\lambda_j}(\max_{i\in A_{js}}\{y_i\} - \mu_j) < \theta_{js}, \end{aligned}$$

Thus, taking $L_{js} = \max\left\{e^{\lambda_j}(\mu_j - \min_{i\in A_{js}}\{y_i\}), e^{-\lambda_j}(\max_{i\in A_{js}}\{y_i\} - \mu_j), 0\right\}$, we can write

$$p(\theta_{js}|\cdots) \propto \begin{cases} \mathrm{Ga}(\theta_{js}|\alpha,\beta_j)\left(\frac{1}{\theta_{js}}\right)^{n_{js}} \mathbb{1}_{(L_{js},\infty)}^{(\theta_{js})} & \text{if } A_{js}\neq\emptyset \\ \mathrm{Ga}(\theta_{js}|\alpha,\beta_j) & \text{if } A_{js}=\emptyset \end{cases} \tag{5.13}$$

- For the allocation variables, see Section 5.2.1.1.

- For the hyper-parameters, the full conditional for ϵ is

$$p(\epsilon|\cdots) \propto \mathbb{1}_{(0,\rho)}^{(\epsilon)}\left(\frac{1}{\epsilon}\right)^{k}\prod_{j=1}^{k}\mathbb{1}_{(\lambda_j)(-\epsilon,\epsilon)}.$$

hence

$$\begin{aligned} 1 &= \prod_{j=1}^{k}\mathbb{1}_{(-\epsilon,\epsilon)}^{(\lambda_j)} \;\Leftrightarrow\; \forall\; j\; |\lambda_j|<\epsilon \;\Leftrightarrow\; \max_{j=1,\ldots,k}\{|\lambda_j|\}<\epsilon, \\ &\Rightarrow \; \mathbb{1}_{(0,\rho)}^{(\epsilon)}\mathbb{1}_{(M_\lambda,\infty)}^{(\epsilon)} = \mathbb{1}_{(M_\lambda,\rho)}^{(\epsilon)} \text{ with } M_\lambda = \max_{j=1,\ldots,k}\{|\lambda_j|\}, \\ &\Rightarrow \; p(\epsilon|\cdots) \propto \left(\frac{1}{\epsilon}\right)^{k}\mathbb{1}_{(M_\lambda,\rho)}^{(\epsilon)}. \end{aligned}$$

For the (β_j) we have the full conditionals as

$$p(\beta_j|\cdots) \propto \mathrm{Ga}\left(\beta_j|a+N\alpha, b+\sum_{s=1}^{N}\theta_{js}\right).$$

It is not straightforward to sample from the full conditionals of our model, and here we describe how to sample from them. First, in (5.11) we need to sample from the density

$$f(x) \propto \mathrm{sech}(x)^m\,\mathbb{1}_{(a,b)}^{(x)}, \tag{5.14}$$

with $m \in \mathbb{N}$, $a, b \in \mathbb{R}$ and $b > a$. For this we have borrowed ideas from Damien and Walker (2001). We introduce the latent variable y, such that if it is integrated out we get back to the original distribution. Specifically, we consider

$$f(x,y) \propto \mathbb{1}_{(a,b)}^{(x)} \mathbb{1}_{(0,(1/\cosh(x))^m)}^{(y)},$$

where we are using the identity $\text{sech}(x) = \cosh(x)^{-1}$. Thus, to sample from (5.14) we generate

$$\begin{aligned} f(y|x) &\propto \mathbb{1}_{(0,(1/\cosh(x))^m)}^{(y)}, \\ f(x|y) &\propto \mathbb{1}_{\left(\max\{a,-\cosh^{-1}(y^{-1/m})\},\min\{b,\cosh^{-1}(y^{-1/m})\}\right)}^{(x)}. \end{aligned}$$

This is implemented trivially due to the fact that $\cosh(x)$ and $\cosh^{-1}(x) = \text{arcosh}\,(x)$ are basic functions available in almost all programming languages.

Second, in (5.13) we need to sample from

$$f(x) \propto x^{\alpha-(n+1)} e^{-\beta x} \mathbb{1}_{(c,\infty)}^{(x)},$$

with $\alpha, \beta > 0, c \geq 0$ and $n \in \mathbb{N}$. We proceed as in the previous case:

$$f(x,y) \propto x^{\alpha-(n+1)} \mathbb{1}_{\left(0,e^{-\beta x}\right)}^{(y)} \mathbb{1}_{(c,\infty)}^{(x)},$$

which leads to

$$\begin{aligned} f(y|x) &\propto \mathbb{1}_{\left(0,e^{-\beta x}\right)}^{(y)}, \\ f(x|y) &\propto x^{\alpha-(n+1)} \mathbb{1}_{(c,-log(y)/\beta)}^{(x)}. \end{aligned}$$

Both are easy to sample.

Finally, the inverse transformation technique can be used to sample from the full conditional of ϵ, and to sample from truncated normal distributions for the locations the slice sampling techniques of Damien and Walker (2001) can be used.

In our implementations, to generate one sample from the truncated distributions the slice sampler was iterated nine times, hence we kept the random variate generated in the

last iteration. Also note that the starting values in every case where chosen as close as possible to the mode of the target distribution. With these considerations we observed good trend between efficiency and performance reducing the correlation and obtaining an efficient sampler.

5.2.2 Unknown number of components

To develop the case for an unknown number of components, we will use the ideas described in Section 2.3. Thus we view reversible jump as a particular case of the product space model discussed by Godsill (2001), and the main objective is to derive the acceptance probability (2.19) to generate a Markov chain with invariant probability distribution $p(\phi^{(k)}, \tau^{(k)}, k|\mathbf{y})$.

Proceeding as in Section 2.3.2, we first identify our variables according to (2.19):

$$\begin{aligned} \phi^{(k)} &= (\mathbf{w}^{(k)}, \boldsymbol{\lambda}^{(k)}, \boldsymbol{\mu}^{(k)}, \boldsymbol{\beta}^{(k)}, \mathbf{v}^{(k)}, \boldsymbol{\theta}^{(k)}) \\ \tau^{(k)} &= (\mathbf{z}, \mathbf{d})^{(k)} \end{aligned}$$

with $\mathbf{w}^{(k)} = \{w_j\}_{j=1}^k$, $\boldsymbol{\lambda}^{(k)} = \{\lambda_j\}_{j=1}^k$, $\boldsymbol{\mu}^{(k)} = \{\mu_j\}_{j=1}^k$ and $\boldsymbol{\beta}^{(k)} = \{\beta_j\}_{j=1}^k$ vectors and $\mathbf{v}^{(k)} = \left\{\{v_{js}\}_{s=1}^N\right\}_{j=1}^k$, $\boldsymbol{\theta}^{(k)} = \left\{\{\theta_{js}\}_{s=1}^N\right\}_{j=1}^k$ matrices of $k \times N$.

5.2.2.1 Hybrid strategy

The Markov chain is constructed via a hybrid strategy (see Appendix A, Section A.4.4), we will have the following moves:

1. For a fixed k, a Gibbs kernel is used; this move was described in Section 5.2.1.3.

2. Split-combine move.

3. Birth-death of empty components.

4. Once the birth-death move has been attempted we return to the first step, hence forming cycle. This is repeated until the sampler has converge.

Remark 5.1. There are two important observations to make: first, the transformation was devised for $\mathbf{v}^{(k)}$ because with the stick-breaking representation $\left\{\{w_{js}\}_{s=1}^{N}\right\}_{j=1}^{k}$ are just functions of them and second, in the trans-dimensional step the slice variables are fixed.

5.2.2.2 Split and combine move

We will follow the split-combine strategy carefully described in Section 2.3.2, hence the ratio to update the model index is given by (2.20).

Remark 5.2. To devise the transformation (2.15) for non-trivial problems is the hardest part in any reversible jump strategy. In our case this was really complicated since the infinite weights are functions of the v's (and we only know the prior over the v's), thus the transformations had to be defined for the v's. We tried several alternatives, here we present the one that delivered the most efficient sampler.

For the split we generate u_1 and u_2 from a $\text{Be}(2,2)$, hence

$$\begin{aligned} w_{j_1} = w_j u_1 \quad &\leftrightarrow \quad w_{j_2} = w_j(1-u_1) \\ \mu_{j_1} = \mu_j - \sigma_j u_2 \sqrt{\frac{1-u_1}{u_1}} \quad &\leftrightarrow \quad \mu_{j_2} = \mu_j + \sigma_j u_2 \sqrt{\frac{u_1}{1-u_1}}, \end{aligned}$$

where

$$\sigma_j^2 = \cosh(\lambda_j)^2 \left(\frac{1}{3} \sum_{s=1}^{N} w_{js} \theta_{js}^2 \right).$$

The location parameters must satisfy $\mu_{j-1} < \mu_{j_1}$ and $\mu_{j_2} < \mu_{j+1}$ ($\mu_0 = -\infty$ and $\mu_{k+1} = \infty$) if not, the move is rejected immediately. The combine is automatically determined as

$$\begin{aligned} w_j = w_{j_1} + w_{j_2} \quad &\leftrightarrow \quad u_1 = \frac{w_{j_1}}{w_{j_1} + w_{j_2}} \\ \mu_j = \mu_{j_1} u_1 + \mu_{j_2}(1-u_1) \quad &\leftrightarrow \quad u_2 = \frac{(\mu_{j_2} - \mu_{j_1})\sqrt{u_1(1-u_1)}}{\sigma_j}. \end{aligned}$$

For the transformation of the skewness and variance we used the skewness and standard deviation of the prior predictive:

$$\begin{aligned}\text{skew}(Y|\lambda,\alpha,\beta) &= \frac{6\sqrt{6}(\alpha+(2+\alpha)\cosh(2\lambda)\sinh(\lambda)}{\sqrt{\alpha}(\alpha-2+(4+\alpha)\cosh(2\lambda))^{3/2}}\\ \text{std}(Y|\lambda,\alpha,\beta) &= \frac{\sqrt{\alpha(\alpha-2+(4+\alpha)\cosh(2\lambda))}}{\sqrt{6}\beta},\end{aligned}$$

where

$$\text{skew}(Y|\lambda,\alpha,\beta) = \frac{\gamma_3(Y|\lambda,\alpha,\beta)}{\text{Var}\,(Y|\lambda,\alpha,\beta)^{3/2}}$$

and $\gamma_l = \mathbb{E}\left((Y-\mathbb{E}(Y))^l\right)$ the l^{th} moment about the mean.

After trying different alternatives we chose

$$\begin{aligned}\sinh(\lambda_j) &= \frac{\sinh(\lambda_{j1})+\sinh(\lambda_{j2})}{2}\\ \text{std}(Y|\lambda_j,\alpha,\beta_j) &= \text{std}(Y|\lambda_{j_1},\alpha,\beta_{j_1})+\text{std}(Y|\lambda_{j_2},\alpha,\beta_{j_2}),\end{aligned}$$

and defining

$$\begin{aligned}u_3 &= \frac{\sinh(\epsilon)+\sinh(\lambda_{j1})}{2\sinh(\epsilon)+\sinh(\lambda_{j1})+\sinh(\lambda_{j2})}\\ u_4 &= \frac{1}{1+\frac{\beta_{j_1}\sqrt{\alpha-2+(4+\alpha)\cosh(2\lambda_{j_2})}}{\beta_{j_2}\sqrt{\alpha-2+(4+\alpha)\cosh(2\lambda_{j_1})}}},\end{aligned}$$

we have the combine for these two parameters already set. For the split we work out $\lambda_{j_1}, \lambda_{j_2}, \beta_{j_1}$ and β_{j_2} generating u_3 and u_4 from a $\text{Be}(2,2)$

$$\begin{aligned}\sinh(\lambda_{j_1}) &= 2u_3\sinh(\lambda_j)-(1-2u_3)\sinh(\epsilon)\\ \sinh(\lambda_{j_2}) &= 2(1-u_3)\sinh(\lambda_j)+(1-2u_3)\sinh(\epsilon)\\ \beta_{j_1} &= \frac{\beta_j\sqrt{\alpha-2+(4+\alpha)\cosh(2\lambda_{j1})}}{u_4\sqrt{\alpha-2+(4+\alpha)\cosh(2\lambda_j)}}\\ \beta_{j_2} &= \frac{\beta_j\sqrt{\alpha-2+(4+\alpha)\cosh(2\lambda_{j2})}}{(1-u_4)\sqrt{\alpha-2+(4+\alpha)\cosh(2\lambda_j)}}.\end{aligned}$$

The full joint of v_{js}, see Algorithm 3.5, is given by

$$\begin{aligned}p(v_{js}|\cdots) &= \text{Be}(v_{js}|1+n_{js},c+m_{js})\\ &\Rightarrow \text{mode}(v_{js}|\cdots) = \frac{n_{js}}{n_{js}+m_{js}+c-1}\end{aligned}$$

with $m_{js} = n_j - \sum_{l=1}^{s} n_{jl}$.

Thus assuming

$$v_{js} = \frac{n_{js}}{n_{js} + m_{js}} \Rightarrow \frac{m_{js}}{n_{js}} = \frac{(1 - v_{js})}{v_{js}}$$

so if $u_{5s} \sim \text{beta}(1, 1)$, for the split, we can think in

$$n_{j_1 s} \approx u_{5s} n_{js} \quad \leftrightarrow \quad n_{j_2 s} \approx (1 - u_{5s}) n_{js}$$

$$m_{j_1 s} \approx u_{*s} m_{js} \quad \leftrightarrow \quad m_{j_2 s} \approx (1 - u_{*s}) m_{js}$$

hence,

$$\begin{aligned} v_{j_1 s} &= \frac{n_{j_1 s}}{n_{j_1 s} + m_{j_1 s}} = \frac{u_{5s} v_{js}}{u_{5s} v_{js} + u_{*s}(1 - v_{js})} \\ v_{j_2 s} &= \frac{n_{j_2 s}}{n_{j_2 s} + m_{j_2 s}} = \frac{(1 - u_{5s}) v_{js}}{(1 - u_{5s}) v_{js} + (1 - u_{*s})(1 - v_{js})}, \end{aligned}$$

for simplicity we imposed $u_{*s} = 1/2$ for all s, so the split will be defined as

$$v_{j_1 s} = \frac{2 u_{5s} v_{js}}{1 - (1 - 2u_{4s}) v_{js}} \quad \leftrightarrow \quad v_{j_2 s} = \frac{2(1 - u_{5s}) v_{js}}{1 + (1 - 2u_{5s}) v_{js}},$$

then we generate $w_{j_1 s}$ and $w_{j_2 s}$ for $s = 1, \ldots, N$ via the stick-breaking representation. The combine is easily calculated

$$v_{js} = \frac{v_{j_1 s} + v_{j_2 s} - 2 v_{j_1 s} v_{j_2 s}}{2 - v_{j_1 s} - v_{j_2 s}} \leftrightarrow u_{5s} = \frac{v_{j_1 s} - v_{j_1 s} v_{j_2 s}}{v_{j_1 s} + v_{j_2 s} - 2 v_{j_1 s} v_{j_2 s}},$$

and again, we generate w_{js} for $s = 1, \ldots, N$ using the stick-breaking representation.

To combine θ_{js} we related

$$\theta_{js}(|\sinh(\lambda_j)| + 1) = \theta j_1 s(|\sinh(\lambda_{j_1})| + 1) + \theta_{j_2 s}(|\sinh(\lambda_{j_2})| + 1),$$

where $\theta_{js}|\sinh(\lambda_j)|$ is the length of the base of sub-component s of the random histogram j. The easiest mapping to $(0, 1)$ is

$$u_{5s} = \frac{\theta_{j_1 s}(|\sinh(\lambda_{j_1})| + 1)}{\theta_{j_1 s}(|\sinh(\lambda_{j_1})| + 1) + \theta_{j_2 s}(|\sinh(\lambda_{j_2})| + 1)}.$$

To split, we generate $u_{6s} \sim \text{beta}(2,2)$ for $s = 1, \ldots, N$ and consider

$$\begin{aligned} \theta_{j_1 s} &= u_{6s}\theta j s \frac{(|\sinh(\lambda_j)|+1)}{(|\sinh(\lambda_{j_1})|+1)} \\ \theta_{j_2 s} &= (1-u_{6s})\theta j s \frac{(|\sinh(\lambda_j)|+1)}{(|\sinh(\lambda_{j_2})|+1)}. \end{aligned}$$

With $T(\phi^{(k)}, \mathbf{u})$ and the auxiliary variables already defined we can calculate the absolute value of the determinant of the Jacobian. Interchanging rows of the Jacobian matrix: block matrices are obtained. Thus, we can use the fact that, if $\mathbf{J}$ is a square block matrix,

$$\mathbf{J} = \begin{pmatrix} \mathbf{A}_{mxm} & \mathbf{0}_{mxn} \\ \mathbf{C}_{nxm} & \mathbf{D}_{nxn} \end{pmatrix}$$

where $\mathbf{0}_{mxn}$ is a matrix of zeroes. Hence, $Det(\mathbf{J}) = Det(\mathbf{A})Det(\mathbf{D})$, and using this fact recursively, we can calculate the ratio between the determinant of the Jacobian and the density of the independent variables in (2.19):

$$\begin{aligned} \frac{1}{q_{2\mathbf{u}}(\mathbf{u})} \left| \frac{\partial T(\phi^{(k)}, \mathbf{u})}{\partial(\phi^{(k)}, \mathbf{u})} \right| &= w_j \frac{\sigma_j}{\sqrt{u_1(1-u_1)}} \\ &\times \frac{4\cosh(\lambda_j)(\sinh(\epsilon)+\sinh(\lambda_j))}{\sqrt{(1+\sinh(\lambda_{j1})^2)(1+\sinh(\lambda_{j2})^2)}} \\ &\times \frac{\beta_{j1}\beta_{j2}}{\beta_j u_3(1-u_3)} \left[\frac{1}{2^N} \prod_{s=1}^{N} \frac{v_{js}\,(2-v_{j_1 s}-v_{j_2 s})^3}{(1-v_{j_1 s})(1-v_{j_2 s})} \right] \\ &\times \left(\frac{(1+|\sinh(\lambda_j)|)^2}{(1+|\sinh(\lambda_{j_1})|)(1+|\sinh(\lambda_{j_2})|)} \right)^N \prod_{s=1}^{N} \theta_{js} \\ &\times \frac{1}{\prod_{l=1}^{4} g_{2,2}(u_l) \prod_{s=1}^{N} g_{1,1}(u_{5s}) g_{2,2}(u_{6s})}. \end{aligned} \tag{5.15}$$

The first line of (5.15) is the product of the determinants of the Jacobian for the weights of the finite mixture and the locations. The second row is the determinant of the Jacobian for the skewness and the third is for the variance and v's The fourth row is the determinant of the Jacobian of the θ's. Finally, the denominator in last line is the product of the density functions of the independent extra random variables: $g_{a,b}(\cdot)$ is the density function of a $\text{Be}(\cdot|a,b)$.

The proposals for the allocation variables are as follows: generate two vectors $\mathbf{b} = \{b_i\}_{i=1}^n$ and $\mathbf{e} = \{e_i\}_{i=1}^n$ such that $b_i = z_i$ and $e_i = d_i$ for $i = 1, \ldots, n$. For the split, first, all the observations such that $b_i = j^*$ with j^* bigger than j are re-allocated to $b_i = j^* + 1$, and leave $\mathbf{e}$ untouched. Second, with the split variables, the observations such that $b_i = j$ must be re-allocated randomly to j_1 or j_2, with probabilities

$$p_{i,l,s} \propto w_{j_l} w_{j_l s} e^s \, \mathrm{U}\left(y_i | - \theta_{j_l s} e^{-\lambda_{j_l}} + \mu_{j_l}, \theta_{j_l s} e^{\lambda_{j_l}} + \mu_{j_l}\right), \tag{5.16}$$

for $l = 1, 2$ and $s \in \{1, 2, \ldots, N_i\}$ (N_i result of the slice variables). Observe that this will generate new allocations for the (e_i) such that $b_i = j$ as well.

With $(\mathbf{b}, \mathbf{e})$ already re-allocated, and using the probabilities (5.16), we calculate the probability for the proposal of the discrete variables in the split, this is given by

$$p(\tau^{(k+1)} | \phi^{(k)}, \tau^{(k)}) = \prod_{\{i: z_i = j\}} p_{i, b_i, e_i}. \tag{5.17}$$

For the combine; we re-allocate the $b_i = j^*$ bigger than or equal to j_2 into $j^* - 1$. Hence, for the allocations such that $b_i = j_1$, e_i must be re-allocated. This is done randomly with probabilities

$$p_{i,s|j_1} = \frac{p_{i,j_1,s}}{p_{i,j_1}}, \text{ for } s = 1, \ldots, N_i, \quad \text{with } p_{i,j_1} = \sum_{s=1}^{N_i} p_{i,j_1,s}$$

where here (5.16) is calculated with the combined random variables.

With $(\mathbf{b}, \mathbf{e})$ re-allocated the probability for the proposal of the discrete variables in the combine is

$$p(\tau^{(k)} | \phi^{(k+1)}, \tau^{(k+1)}) = \prod_{\{i: z_i = j_1\}} p_{i, e_i | j_1}. \tag{5.18}$$

Then, in (2.19) we substitute the values of (5.17) and (5.18).

In this case we do not have the advantage of the deterministic move for the allocations in the combine, as with the mixture of normals. We need to calculate (5.17) and (5.18) in both, split and combine moves.

To complete the specification of (2.19) we only need to calculate

$$\begin{aligned}
\frac{p(\phi^{(k+1)},\tau^{(k+1)},k+1|\mathbf{y})}{p(\phi^{(k)},\tau^{(k)},k|\mathbf{y})} &= \frac{\prod_{i\in E_{j_1}\cup E_{j_2}} \mathrm{U}\left(y_i|\mu_{b_i}-\theta_{b_ie_i}e^{-\lambda_{b_i}},\mu_{b_i}+\theta_{b_ie_i}e^{\lambda_{b_i}}\right)}{\prod_{i\in A_j} \mathrm{U}\left(y_i|\mu_j-\theta_{jd_i}e^{-\lambda_j},\mu_j+\theta_{jd_i}e^{\lambda_j}\right)} \\
&\times \frac{p(k+1)}{p(k)}\frac{w_{j_1}^{n_{j_1}+\delta-1}w_{j_2}^{n_{j_2}+\delta-1}}{w_j^{n_j+\delta-1}B(\delta,k\delta)}\left(\frac{1}{2\epsilon}\right) \\
&\times \prod_{s=1}^{N}\frac{w_{j_1s}^{n_{j_1s}}w_{j_2s}^{n_{j_2s}}}{w_{js}^{n_{js}}}\exp(-\sum_{i=1}^{n}(d_i^- - d_i^+))(k+1)\frac{1}{\sigma_0\sqrt{2\pi}} \\
&\times \exp(-\frac{1}{2\sigma_0^2}\left((\mu_{j_1}-\mu_0)^2+(\mu_{j_2}-\mu_0)^2-(\mu_j-\mu_0)^2\right)) \\
&\times \frac{b^a}{\mathrm{Ga}(a)}\left(\frac{\beta_{j_1}\beta_{j_2}}{\beta_j}\right)^{a-1}\exp(-b(\beta_{j_1}+\beta_{j_2}-\beta_j)) \\
&\times \left(\frac{\beta_{j_1}\beta_{j_2}}{\beta_j}\right)^{N\alpha}\left(\frac{1}{\Gamma(\alpha)}\right)^N\prod_{s=1}^{N}\left(\frac{\theta_{j_1s}\theta_{j_2s}}{\theta_{js}}\right)^{\alpha-1} \\
&\times \exp(-(\beta_{j_1}\theta_{j_1s}+\beta_{j_2}\theta_{j_2s}-\beta_j\theta_{js})) \\
&\times \left(\frac{1}{B(1,c)}\right)^N\prod_{s=1}^{N}\left(\frac{(1-v_{j_1s})(1-v_{j_2s})}{(1-v_{js})}\right)^{c-1}.
\end{aligned}$$

where $E_{j_1}=\{i:b_i=j_1\}$ and $E_{j_2}=\{i:b_i=j_2\}$.

5.2.2.3 Birth and death move

The birth-death move of empty components is as in Section 2.3.2. In this case, for the birth, a weight and parameters for the proposed new component are drawn using

$$w_{j^*}\sim \mathrm{Be}(1,k),\quad \lambda_{j^*}\sim \mathrm{U}(-\epsilon,\epsilon),\quad \mu_{j^*}\sim \mathrm{N}(\mu_0,\sigma_0^2),\quad \beta_{j^*}\sim \mathrm{Ga}(a,b)\ \text{ and}$$

$$v_{j^*s}\sim \mathrm{Be}(1,c)\ \text{ and }\ \theta_{j^*s}\sim \mathrm{Ga}(\alpha,\beta_{j^*}),\quad \text{for } s=1,\ldots,N_i.$$

Hence, the independent variable is given by

$$\mathbf{u}=(w_{j^*},\lambda_{j^*},\mu_{j^*},\beta_{j^*},v_{j^*1},\ldots,v_{j^*N_i},\theta_{j^*1},\ldots,\theta_{j^*N_i}).$$

5.2.3 Model priors

When working with an unknown number of components, a prior for k is needed and under the assumption of no additional information, for us, the best option is to impose

a discrete uniform prior: $p(k) = \frac{1}{k_{max}} \mathbb{1}_{(1,\ldots,k_{max})}^{(k)}$ for a preselected $k_{max} \in \mathbb{N}$.

Some authors that have worked with normal mixtures suggest a truncated Poisson distribution ($\text{Po}(k|1)\,\mathbb{1}_{(1,\ldots,k_{max})}^{(k)}$), see for example Nobile and Fearnside (2007). This is with the idea of to penalize the presence of empty components, and thus higher values for k. But as we have seen, high overall posterior estimates for k occur due to the use of the normal distribution, so we believe we do not need to follow this idea. All the other priors are as in Section 5.2.1.2.

5.3 Illustrations

5.3.1 Setting the priors

In this section, we describe how to set the unspecified constants of the model (see Section 5.2.1.2). Let R be the range of the data. For the locations we set $\mu_0 = y_{(1)} + R/2$, $\sigma_0^2 = R^2$ and for the finite weights we take $\delta = 1$, giving a uniform prior over the space $w_1 + \cdots + w_k = 1$. For the smoothing parameter of the Dirichlet process we found that results based on $c = 2$ were good, obtaining smooth predictive densities.

The degree of asymmetry allowed is determined by ρ, which bounds the interval where λ can move. We set $\rho = 0.5$ giving room for asymmetry without supporting high variance due to large values of λ; compare the graphics d) and f) with graphic e) in Figure 5.1. For the kurtosis parameter, α, we chose different values depending on the data set to analyze. For example, fixing $\alpha = 2.5$ we obtained good results when estimating a normal distribution, while to estimate a Laplace distribution, $\alpha = 1.01$ was a better option. This will be shown in the next Section. For the remaining unspecified constants we will follow similar ideas to those of Richardson and Green (1997), see Section 2.3.2.1, to devise a "default" prior using the data.

If the asymmetry parameter is zero, the variance of the unimodal distribution is

$$\sigma^2 = \frac{1}{3}\sum_{s=1}^{\infty} w_s \theta_s^2,$$

so proceeding as in Richardson and Green (1997) we can first relate σ to the relevant parameters and then to the range of the data.

To obtain a more practical relationship we calculate

$$\mathbb{E}\left(\sigma^2\right) = \frac{\alpha(\alpha+1)}{3\beta^2} \quad \Rightarrow \quad \sigma \cong \frac{1}{\beta}\sqrt{\frac{\alpha(\alpha+1)}{3}}, \tag{5.19}$$

because $\mathbb{E}(\theta_s^2) = \dfrac{\alpha(\alpha+1)}{\beta^2}$ and $\displaystyle\sum_{s=1}^{\infty} \mathbb{E}(w_s) = \frac{1}{c}\sum_{s=1}^{\infty}\left(\frac{c}{c+1}\right)^s = 1$ (geometric series). Also, θ_s and w_s are independent by construction.

Observe that (5.19) is the variance of the prior predictive (5.5), and could have been obtained alternatively in this manner.

Now note that $\beta \sim \text{Ga}(a,b) \Rightarrow 1/\beta \sim \text{Inv-Ga}(a,b)$, and thus

$$\begin{aligned}
\mathbb{E}(1/\beta) &= \frac{b}{a-1} \quad \text{for} \quad a>1 \;\Rightarrow\; \sigma \cong \frac{b}{a-1}\sqrt{\frac{\alpha(\alpha+1)}{3}}, \\
\text{var}(1/\beta) &= \frac{b^2}{(a-2)(a-1)^2} \quad \text{for} \quad a>2 \;\Rightarrow\; \text{var}(\sigma) \cong \frac{b^2\alpha(\alpha+1)}{3(a-2)(a-1)^2}.
\end{aligned}$$

To connect σ with α, a, b and the data, we solve the system of equations:

$$\begin{cases} \dfrac{b}{a-1}\sqrt{\dfrac{\alpha(\alpha+1)}{3}} &= p_1 R, \\ \dfrac{b^2\alpha(\alpha+1)}{3(a-2)(a-1)^2} &= p_2 R^2, \end{cases}$$

for the variables a and b, where $0 < p_1, p_2 < 1$.

The solution is given by

$$a = 2 + \frac{p_1^2}{p_2} \text{ and } b = \frac{p_1(p_1^2+p_2)}{p_2} R\sqrt{\frac{3}{\alpha(\alpha+1)}}.$$

Hence, setting $p_1 = 0.2$ and $p_2 = 0.5$ we are leaving the Dirichlet process to take care of the tails of each component while being weakly informative about the size of σ.

We do not claim that these choices are non-informative. It is well known there is no way to be fully non-informative under a Bayesian mixture modeling set-up. In fact, it is

well known that the posterior for k is sensitive to the choice of the prior for the location parameters and it is clear that it is heavily influenced by the choice of prior over the variances; see for example the discussions in Richardson and Green (1997), pp.747-749, and Jasra, Holmes, and Stephens (2005), pp. 64-65.

Another way to set the priors for the unimodal distribution is to use an interactive plot of the scaled prior predictive (5.5): $w\,\mathbb{E}\left(f_G(y|\lambda,\mu)\right)$, where $0 < w \leq 1$ and in the background a histogram of the data (we generated our interactive graphic in R using the library `tcltk`, see R Core Team (2012)). The goal here is to obtain a reasonable value to set ρ, α and a sensible knowledge of the range in which β should lie. We set $k_{max} = 30$ in all examples.

5.3.2 Predictive density estimates

To test the unimodal distribution (5.4) we draw samples from six unimodal distributions, namely: Normal; N(0,1), Student's-t; $\mathrm{t}_{(2)}$, Gamma; Ga$(2,3)$, Laplace 1; Lap$(0,20,1)$, Laplace 2; Lap$(0,1,2)$ and Pearson IV: PearsonIV$(3.94, 4.35, 1, 3.74)$. In each case a sample of size $n = 150$ was generated. Since the objective is to assess the range of shapes the unimodal distribution can approximate, by comparing the predictive density with the true density, the samples were not randomly drawn. To improve comparability, we generated a grid of 150 equally spaced points in $(0,1)$ and then evaluated the quantile function, of each distribution, on every point of the grid. For the skew Laplace and Pearson IV distribution, to calculate the quantiles, the libraries `PearsonDS` and `LaplacesDemon` of R were used, see Byron (2012) and Becker (2012).

For the six unimodal data sets, we ran our fixed k sampler, setting $k = 1$, for 200,000 iterations. The first 100,000 iterations were used as a burn-in period. In each case a predictive density estimate was generated (see Appendix 5.B), and these are displayed in Figure 5.2, along with the true density. For the kurtosis parameter, we tried different values and kept the one that gave the best fit. For the remaining parameters we set the

priors as described in the previous section.

From the graphics in Figure 5.2 we see, first, that the predictive densities capture the correct skewness for both symmetric and asymmetric distributions. Second, in all cases, smooth predictive density estimates are obtained. This indicates that the choice $c = 2$, for the smoothing parameter of the Dirichlet process is a sensible choice. Third, for different values of α, different degrees of kurtosis are obtained and this agrees with what was mentioned in Section 5.1.1.1. Looking back to graphic c) in Figure 5.1, the prior predictive suggested that $\alpha = 3.8$ was a good option to generate the density estimate for the standard normal. But we present the predictive density estimate generated with $\alpha = 2.5$ instead. The reason is that the predictive generated with $\alpha = 3.8$ gave a good fit for the tails of the distribution, but produced a flat density estimate.

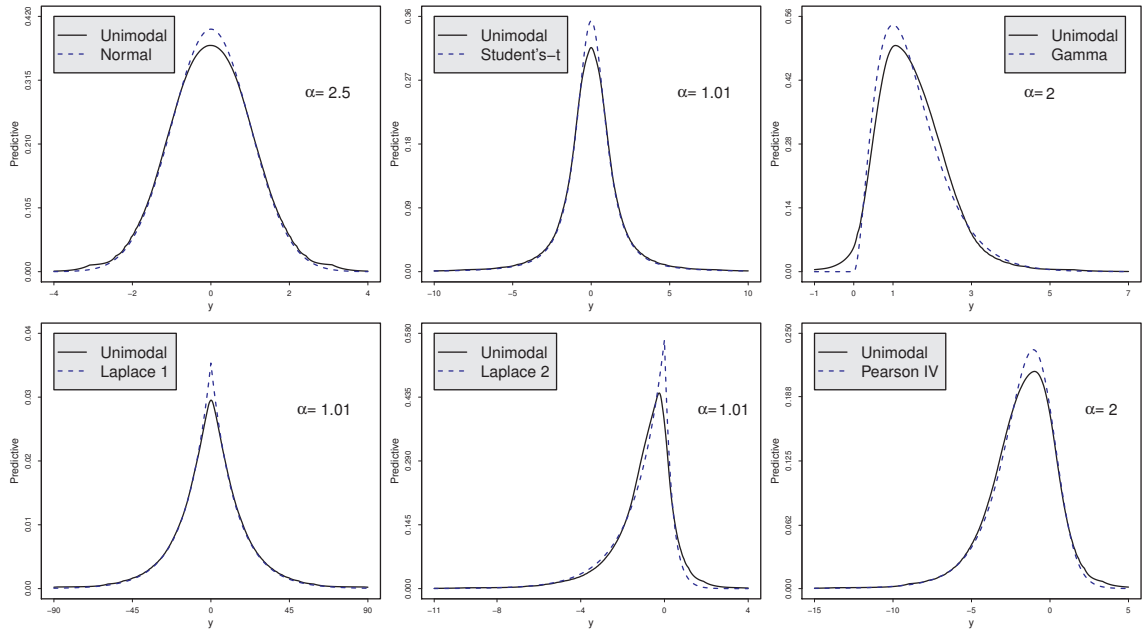

Figure 5.2: Density estimates for six unimodal distributions.

5.3.3 Examples for an unknown number of components

To test the mixture of unimodal distributions (5.6), simulated and real data sets were analyzed. For the simulated data, mixtures of gamma, skew Laplace and skew normal

distributions were used, namely:

Model 1: $0.2\,\mathrm{Ga}(40,20) + 0.6\,\mathrm{Ga}(6,1) + 0.2\,\mathrm{Ga}(200,20)$

Model 2: $0.2\mathrm{Lap}(-5,1,.5) + 0.4\mathrm{Lap}(0,1,1) + 0.3\mathrm{Lap}(3,1,1) + 0.1\mathrm{Lap}(10,1,2).$

Model 3: $0.1\mathrm{SN}(-30,3,-4) + 0.1\mathrm{SN}(-20,3,0) + 0.15\mathrm{SN}(-10,2,4) + 0.15\mathrm{SN}(0,2,-2)$
$+0.1\mathrm{SN}(10,2,3) + 0.1\mathrm{SN}(15,2,2) + 0.1\mathrm{SN}(20,2,4) + 0.1\mathrm{SN}(30,2,0)$
$+0.1\mathrm{SN}(35,2,1).$

The sample size for Models 1 and 2 was $n = 400$ observations and for Model 3 $n = 600$ observations. Model 1 was used by Wiper, Rios-Insua, and Ruggeri (2001) to demonstrate the performance of their reversible jump algorithm for mixtures of gamma distributions. A plot of Models 1 to 3 is shown in Figure 5.3.

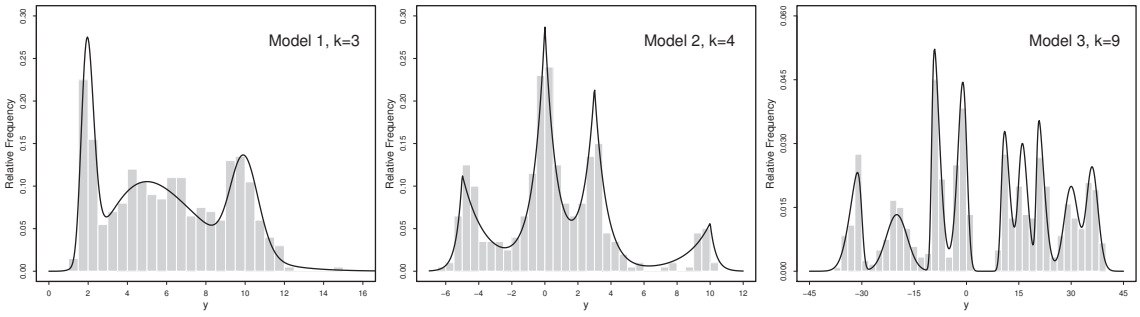

Figure 5.3: Model 1; mixture of gamma distributions, Model 2; mixture of skew Laplace distributions and Model 3 mixture of skew normal distributions.

For the real data sets, we use the same examples used by Richardson and Green (1997): The enzyme and acidity data where described in Chapter 4. The Galaxy data consists of the velocities (in 1000 km/sec) of 82 distant galaxies diverging from our own from six well separated conic sections of the Corona Borealis region. This data set was first studied by Postman, Huchra, and Geller (1986) and it is widely used in the literature to illustrate methodology for mixture modeling. The three data sets are available at the library `mixAK` of R as `Galaxy`, `Acidity` and `Enzyme`. This package contains a variety of statistical methods including MCMC methods to analyze the data

using normal mixtures; see Komárek (2009).

We ran our hybrid sampler for 1,000,000 iterations, the first 200,000 as a burn-in period. Our starting point for k in all the runs was $k = 30$. With the same data sets, we also ran the algorithm described by Richardson and Green (1997), for a mixture of normals model with an unknown number of components, using their default priors (see Section 2.3.2). The same number of iterations and burn-in period was also used.

Posterior probabilities for k for all data sets are given in Table 5.1. For an easy comparison with the mixture of normals, the maximum posterior probabilities for k obtained with the mixture of normals algorithm of Richardson and Green (1997) are presented in Table 5.2. To show that the computational cost of our algorithm is not prohibitive, we present a comparison of CPU times also in Table 5.2.

Table 5.1: Unimodal mixture: posterior distribution of k for the seven data sets, default priors and taking $\alpha = 2.5$.

Data set	n	$p(k\|\mathbf{y})$			
Data from	400	$p(1) = 0.000$	$p(2) = 0.049$	$p(3) = 0.226$	$\boxed{p(4) = 0.280}$
Model 1		$p(5) = 0.211$	$p(6) = 0.126$	$\sum_{k>6} p(k) = 0.108$	
Data from	400	$\sum_{k<4} p(k) = 0.000$	$p(4) = 0.092$	$p(5) = 0.195$	$\boxed{p(6) = 0.227}$
Model 2		$p(7) = 0.196$	$p(8) = 0.133$	$\sum_{k>8} p(k) = 0.156$	
Data from	600	$\sum_{k<7} p(k) = 0.000$	$p(7) = 0.122$	$p(8) = 0.259$	$\boxed{p(9) = 0.281}$
Model 3		$p(10) = 0.190$		$\sum_{k>10} p(k) = 0.149$	
Enzyme	245	$p(1) = 0.000$	$p(2) = 0.257$	$\boxed{p(3) = 0.325}$	$p(4) = 0.222$
		$p(5) = 0.112$	$p(6) = 0.052$	$\sum_{k>6} p(k) = 0.032$	
Acidity	155	$p(1) = 0.000$	$p(2) = 0.149$	$\boxed{p(3) = 0.238}$	$p(4) = 0.235$
		$p(5) = 0.173$	$p(6) = 0.105$	$\sum_{k>7} p(k) = 0.1$	
Galaxy	82	$p(1) = 0.004$	$p(2) = 0.057$	$p(3) = 0.154$	$\boxed{p(4) = 0.217}$
		$p(5) = 0.212$	$p(6) = 0.161$	$\sum_{k>6} p(k) = 0.195$	

In almost all experiments, the posterior distribution for k calculated with the unimodal mixture, supports lower values for k when compared to the normal mixture. There

are only two cases when both models are similar in terms of k: the acidity data and the data from Model 2. Finally, we see that for Model 3, the mixture of skew normal distributions, the posterior calculated via the unimodal distribution gives support to low values for k, the maximum is achieved at $k = 9$, which is the true value. On the other hand, with the normal mixture, the maximum for $p(k|\mathbf{y})$ is attained at $k = 12$.

A comparison of predictive densities is shown in Figure 5.5. These were calculated with the output of the algorithms for a moving k, see Appendix 5.B. It is interesting to note that without giving support to unusually high number of components, in the posterior for k, the unimodal mixture gives accurate representations of each data set. This is an appealing characteristic of our model.

After starting the chains at $k_{max} = 30$ they all moved rapidly to a neighborhood close to the highest posterior value for k. In Figure 5.4 we display the trace for k for the galaxy, acidity and enzyme data (after discarding the burn-in period). They show that in each case the MCMC algorithm mixes well over k. The combined acceptance rates for the two moves that alter the number of components (split-combine and birth-death) were approximately of 15%, 13% and 4% for the galaxy, acidity and enzyme data respectively.

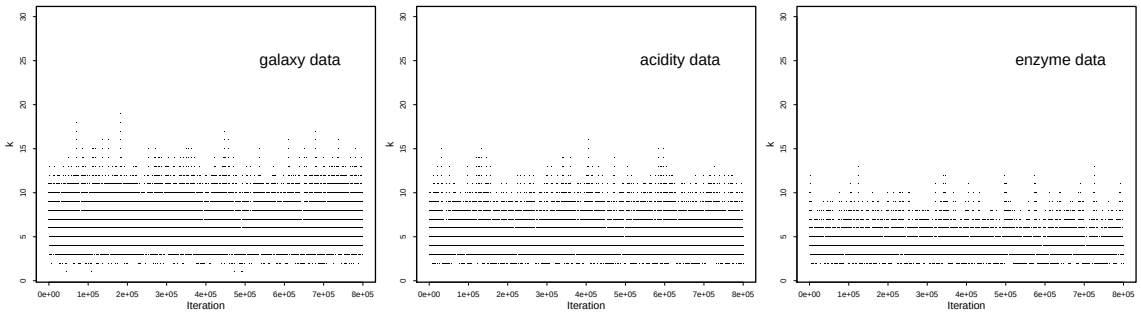

Figure 5.4: Trace for k: galaxy, acidity, and enzyme data.

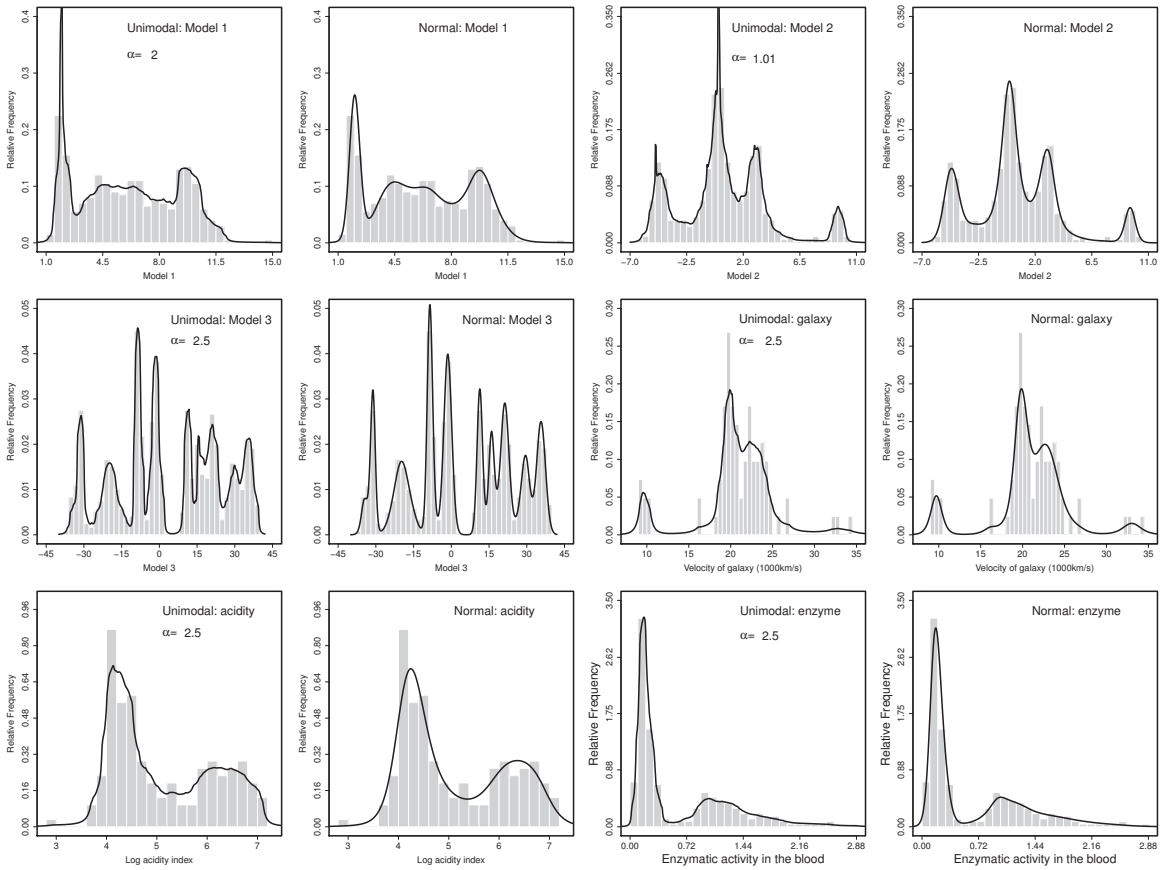

Figure 5.5: Comparison of predictive densities: unimodal vs normal mixture models, default priors.

Table 5.2: True k (if known), normal mixture: maximum posterior distribution of k (default priors) and time comparison between unimodal and normal mixture algorithms.

		normal mixture	approx time (minutes)	
Data set	True k	$\max\{p(k\|\mathbf{y})\}$	normal	unimodal
Model 1	3	$p(5) = 0.260$	8	15
Model 2	4	$p(6) = 0.191$	10	20
Model 3	9	$p(12) = 0.192$	27	29
Enzyme	-	$p(4) = 0.339$	4	9
Acidity	-	$p(3) = 0.258$	3	8
Galaxy	-	$p(6) = 0.189$	3	7

5.3.4 Examples for a known number of components

To make inference for individual components and classification, we need to "undo the label switching", see Chapter 4. With this aim we use the Data relabeling algorithm, see Section 4.3.

We ran the fixed k algorithms, with default priors, for the unimodal and normal mixture. We used the galaxy, acidity and enzyme data assuming, for the unimodal mixture, $k = 4$, $k = 3$ and $k = 3$, respectively, and $k = 6$, $k = 3$ and $k = 4$, for the normal mixture. Note that these are the values for which the posterior distributions attained its maximum for each algorithm. To test our model when the number of underlying components is large, we performed the same comparison using model 3. In all cases, we generated 200,000 iterations, throwing the first 100,000 iterations as a burn-in period. Then we post-processed the MCMC output using the Data relabeling to undo the label-switching and estimate the scaled densities (see Appendix 5.B) and single best clustering. The results are displayed in Figure 5.6. For Model 3 the results are displayed in Figure 5.7, here a plot of the scaled densities and single best clustering calculated via the true model was included.

For the acidity data, the estimated scaled densities and single best clustering obtained

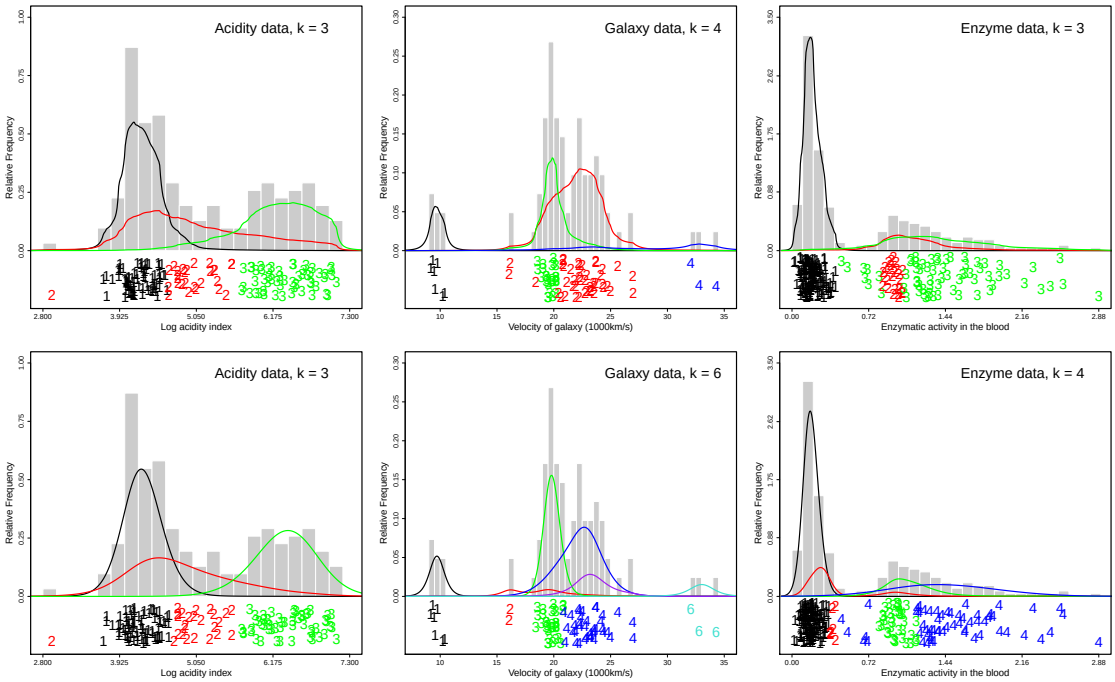

Figure 5.6: Unimodal (top row) and normal mixture for a known k: estimated scaled densities and single best clustering.

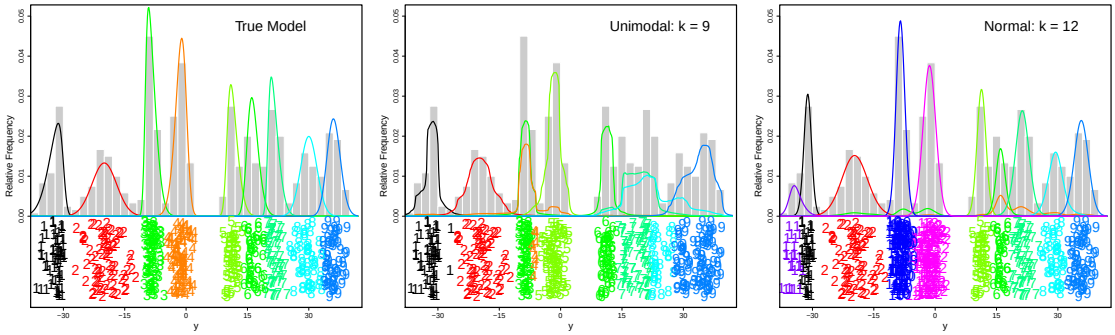

Figure 5.7: True, estimated scaled densities and single best clustering for Model 3.

with the unimodal and normal mixtures are similar. However, from the normal perspective, where a cluster should be a set of observations adequately modeled by a normal distribution, we observe skew scaled densities which makes no sense. See also Figure 4.3. in Chapter 4, the same behavior can be seen when the ECR and KL algorithms are used to undo the label switching. For the galaxy data, in the case of the unimodal mixture, we see four unimodal clusters where each cluster has at least one observation assigned to it. Instead, in the normal case, cluster five has no observations assigned. But beyond this, and more importantly, we observe again a small degree of skewness in clusters three and four of the mixture of normals. For the enzyme data, with the unimodal mixture, from the single best clustering we are able to identify three skew clusters. Instead, to model skewness, the normal mixture needs two normals to model a single cluster (see the cluster labeled as cluster one). For the data set from Model 3, the single best clustering calculated with the unimodal mixture allocates observations to eight clusters, missing group six. There are problems when the components are overlapped, and this can be seen from components three to five. With the normal mixture, the observations are allocated to ten clusters instead of twelve. Here again it is clear that in presence of skew components two normals are needed to model a single skew component, this can be seen in clusters one and eleven for example.

5.3.5 Computational issues

All the experiments were performed in a Dell Precision M4400 (processor Intel core 2 Duo at 2.26 GHz) running Linux openSUSE 12.1. We coded our approach in **C** and used the `.C` Interface to R to have a friendly data input interface, see Peng and Leeuw (2002) for a good introduction. To manage the random seed and generate the random variates from the common distributions we used the GSL-GNU Scientific Library, see Galassi et al. (2009). The analysis and graphics were done in R. All the reported CPU times were measured using the function `system.time` of R.

5.4 Discussion

5.4.1 Conclusions

We have extended the mixture model of Richardson and Green (1997) to allow for clusters to be modeled by unimodal distributions. In all the examples considered, we obtained accurate representations of the data, without giving support to unusually high number of components. This is an appealing feature of the model and provides k with an explicit interpretation: the number of clusters modeled by unimodal densities.

In the absence of further information, it is natural to associate the number of clusters with the number of unimodal distributions. Hence, if we assume unimodality instead of normality for the components distribution of a mixture, we are giving a proper meaning to k. For this, obviously, $f_G(y|\lambda, \mu)$ must be unimodal, which is based on Feller's representation of unimodal and symmetric distributions.

It is fair to say that replacing the nonparametric density (5.4) with a flexible parametric family, which includes skewness and kurtosis parameters, would result in a simpler model. However, we believe that this would bring problems for modeling tails. A parametric model carries a certain type of heavy tail and given the nature of the problem we are tackling, tails will be playing an important role. It is imperative not to get them wrong, and parametric models offer this opportunity. Nonparametric models do not.

We believe that the ideas of Godsill (2001) nicely complement the paper of Green (1995) and allow us to think clearly about how to deal with a trans-dimensional problems. We also believe that we would not have been able to implement an MCMC strategy without the use of the slice variables needed to sample each $(f_{G_j}(y|\lambda_j, \mu_j))$. These two concepts together give us the ability to move a stochastic process across sub-spaces of different dimensions.

5.4.2 Future Work

One of the aims is to extend our ideas to regression models and also a multivariate model. For the latter we have to construct a nonparametric unimodal and multivariate distribution, which is not so straightforward. The initial idea is to describe a multivariate unimodal distributions via $Y_j = U_j Z_j$, for $j = 1, \ldots, p$, where the $U = (U_1, \ldots, U_p)$ are i.i.d standard normal and the $Z = (Z_1, \ldots, Z_p)$ have any joint distribution. We would be trying to extend Khinchin's characterization (Khinchin (1938)) of univariate unimodal distributions to the multivariate setting, see Devroye (1997) and Tao (1989).

From the four parameter densities that can be found in the literature: the Pearson IV family of densities and the densities summarized in Rigby and Stasinopoulos (2005), pp. 516-517, which includes the Box-Cox power exponential among others, we believe that the prior predictive (5.5) deserves further study. Its closed form is easy to write in full and its parameters can be easily understood. Here the idea would be to analyze this unimodal distribution further: find efficient methods to generate random variates from it, estimate its parameters (moments, maximum likelihood and posterior mean estimates), obtain (if possible) the normal distribution or other standard distributions from it (or a version of it), etc. Also, it would be of interest to use (5.5) to model the residuals in a regression model or as the components distribution in a finite mixture model.

5.A Kernel

A non-negative valued function $k(y|\theta)$ defined on $(\mathbb{R}^2, \mathscr{B}(\mathbb{R}^2))$ is a valid kernel iff

$$\int_{\mathbb{R}} k(y|\theta)dy = 1 \text{ for each } \theta \in \mathbb{R}, \tag{5.20}$$

$$\int_{\mathbb{R}} k(y|\theta)G_0(d\theta) < \infty \text{ for each } y \in \mathbb{R}, \tag{5.21}$$

where G_0 is a distribution function, see Lo (1984).

We use the function

$$k(y|\theta) = \mathrm{U}(y| - \theta, \theta)\, \mathbb{1}_{(0,\infty)}^{(\theta)} = \frac{1}{2\theta}\, \mathbb{1}_{(-\theta,\theta)}^{(y)}\, \mathbb{1}_{(0,\infty)}^{(\theta)}, \tag{5.22}$$

note that $\mathbb{1}_{(-\theta,\theta)}^{(y)}\, \mathbb{1}_{(0,\infty)}^{(\theta)} = \mathbb{1}_{(|y|,\infty)}^{(\theta)}$, and set G_0 as the gamma distribution function with density

$$g_0(\theta|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)}\theta^{\alpha-1}e^{-\beta\theta}\, \mathbb{1}_{(0,\infty)}^{(\theta)} \quad \alpha > 0, \beta > 0.$$

All the conclusions remain valid, with small modifications, for the function

$$k(y|\theta,\lambda,\mu) = \mathrm{U}\left(y| - \theta e^{-\lambda} + \mu, \theta e^{\lambda} + \mu\right) \mathbb{1}_{(0,\infty)}^{(\theta)}$$

but for ease of notation we will stay with (5.22).

It is straightforward to see that $k(y|\theta)$ is a non-negative function and satisfies condition (5.20). However it is not clear that it satisfies condition (5.21), hence we need to show that for every $y \in \mathbb{R}$

$$\int_{\mathbb{R}} k(y|\theta)G_0(d\theta|\alpha,\beta) = \int_{|y|}^{\infty} \frac{\beta^\alpha}{2\Gamma(\alpha)}\theta^{\alpha-2}e^{-\beta\theta}d\theta < \infty$$

It is clear that for each $y \in \mathbb{R} \setminus \{0\}$

$$\begin{aligned} 0 \le \int_{|y|}^{\infty} \frac{\beta^\alpha}{2\Gamma(\alpha)}\theta^{\alpha-2}e^{-\beta\theta}d\theta &\le \int_{0}^{\infty} \frac{\beta^\alpha}{2\Gamma(\alpha)}\theta^{\alpha-2}e^{-\beta\theta}d\theta \\ &= \frac{\beta^\alpha}{2\Gamma(\alpha)}\left(\frac{\theta^{\alpha-1}}{\alpha-1}e^{-\beta\theta}\Big|_{\theta=0}^{\infty} + \frac{\beta}{\alpha-1}\int_0^{\infty}\theta^{\alpha-1}e^{-\beta\theta}d\theta\right) \\ &= \frac{\beta^\alpha}{2\Gamma(\alpha)(\alpha-1)}\theta^{\alpha-1}e^{-\beta\theta}\Big|_{\theta=0}^{\infty} + \frac{\beta}{2(\alpha-1)}. \end{aligned} \tag{5.23}$$

We see that for $\alpha = 1$ (5.23) is not defined and for $\alpha < 1$ it goes to $-\infty$ which makes no sense. Hence to have a valid kernel we need to impose the constraint $\alpha > 1$, in this case (5.23) is equal to $\frac{\beta}{2(\alpha-1)} < \infty$.

Now if we define

$$w(y) = \int_y^{\infty} \frac{\beta^\alpha}{2\Gamma(\alpha)}\theta^{\alpha-2}e^{-\beta\theta}d\theta \;\text{ and }\; a(y) = |y|$$

a simplified version of our prior predictive,(5), $\mathbb{E}(f_G(y))$ with the kernel (5.22) can be set: $g(y) = w(a(y)) = \mathbb{E}(f_G(y))$. Thus to find the range of values where $\mathbb{E}(f_G(y))$ is

differentiable (smooth) the chain rule can be used: $g'(y) = -a'(y)\frac{\beta^\alpha}{2\Gamma(\alpha)}|y|^{\alpha-2}e^{-\beta|y|}$ where $a'(y) = 1$ if $y \geq 0$ and $a'(y) = -1$ otherwise.

Clearly if $1 < \alpha < 2$, then

$$\lim_{y\to 0^-} g'(y) = \infty \text{ and } \lim_{y\to 0^+} g'(y) = -\infty,$$

for $\alpha = 2$ we have

$$\lim_{y\to 0^-} g'(y) = \frac{\beta^2}{2} \text{ and } \lim_{y\to 0^+} g'(y) = -\frac{\beta^2}{2},$$

and for $\alpha > 2$, $\lim_{y\to 0} g'(y) = 0$.

In summary, first to have a valid kernel we need to impose $\alpha > 1$. Second, at $y = 0$ the prior predictive $\mathbb{E}(f_G(y))$ is a continuous function if $1 < \alpha \leq 2$ but the derivative does not exist. Third, for a smooth prior predictive at $y = 0$, $\alpha > 2$ is required. For the kernel $k(y|\theta, \lambda, \mu)$, we take $a(y) = \max\{(\mu - y)e^\lambda, (y - \mu)e^{-\lambda}\}$ so we will have the same three conclusions. In this case the point where the derivative does not exist is $y = \mu$.

5.B Predictive density estimate

At iteration t of the hybrid MCMC sampler the following values are generated

$$k^t, N^t, \{w_j^t\}_{j=1}^{k^t}, \{\lambda_j^t\}_{j=1}^{k^t}, \{\mu_j^t\}_{j=1}^{k^t}, \{\beta_j^t\}_{j=1}^{k^t},$$
$$\left\{\{w_{js}^t\}_{s=1}^{N^t}\right\}_{j=1}^{k^t} \text{ and } \left\{\{\theta_{js}^t\}_{s=1}^{N^t}\right\}_{j=1}^{k^t}$$

where k^t and N^t are the number of components of the finite mixture and the truncation point of the infinite mixture, (5.9), respectively and if k is fixed, then $k^t = k$ for all t.

Hence for $\{w_{js}^t\}_{s=1}^{N^t}$ to sum one for each j, we need to generate

$$w_{jN^t+1}^t = 1 - \sum_{s=1}^{N^t} w_{js}^t \text{ and } \theta_{jN^t+1}^t \sim \text{Ga}(\alpha, \beta_j^t).$$

Thus the predictive for a new observation y^*, conditionally independent to $\mathbf{y} = \{y_i\}_{i=1}^n$, is approximated via

$$p(y^*|\mathbf{y}) \approx \frac{1}{M}\sum_{t=1}^{M}\sum_{j=1}^{k^t} w_j^t \sum_{s=1}^{N^t+1} w_{js}^t \,\text{U}\left(y^*|\mu_j^t - \theta_{js}^t e^{-\lambda_j^t}, \mu_j^t + \theta_{js}^t e^{\lambda_j^t}\right)$$

where M is the number of iterations after the burn-in period.

For a fixed k, to approximate the scaled density for component j, we first undo the label switching and then calculate

$$\frac{1}{M}\sum_{t=1}^{M} w_j^t \sum_{s=1}^{N^t+1} w_{js}^t \,\mathrm{U}\left(y^* | \mu_j^t - \theta_{js}^t e^{-\lambda_j^t}, \mu_j^t + \theta_{js}^t e^{\lambda_j^t}\right).$$

Appendix A

MCMC Background

A.1 Motivation

Under the Bayesian setting, in addition to specifying the model for the observed data $\mathbf{y} = \{y_i\}_{i=1}^n$ given a vector of unknown parameters θ, usually in the form of a probability distribution $p(\mathbf{y}|\theta)$, we treat θ as a random variable. Then, to make inference about θ the idea is to update our prior beliefs, expressed via $p(\theta)$, using Bayes' rule:

$$p(\theta|\mathbf{y}) = \frac{p(\mathbf{y}|\theta)p(\theta)}{\int_\Theta p(\mathbf{y}|\theta)p(\theta)\mathbf{d}\theta}. \tag{A.1}$$

This can be seen as a learning process: first, we set a model for the data (as in the classic framework); second, we introduce our prior beliefs into the problem via $p(\theta)$; and third, once we have observed the data our knowledge about θ is updated calculating the posterior distribution (A.1).

Then we can calculate posterior summaries such as

$$\mathbb{E}(g(\theta)|\mathbf{y}) = \int_\Theta g(\theta)p(\theta|\mathbf{y})\mathbf{d}\theta, \tag{A.2}$$

as sensible choice for a point estimate of θ, or if we want a Bayesian confidence interval we can find the $\alpha/2$ and $(1-\alpha/2)$ quantiles of $p(\theta|\mathbf{y})$, see Carlin and Louis (2000) for a complete treatment on Bayesian methods.

The fact is that only in trivial cases it is possible to work out (A.1) analytically to then calculate (A.2) exactly. However, with the advent of Markov chain Monte Carlo (MCMC) methods it is now possible find ways to generate samples with the property that the empirical distribution of the sample approximates the posterior distribution (A.1) and using such samples it is easy to estimate integrals such as (A.2) or any other posterior summaries.

In this Appendix we summarize the results of Markov chain theory and the basic ideas under some MCMC strategies as used in this thesis: Gibbs sampler, Metropolis-Hastings and slice sampler. The material of this Appendix is based on general state space Markov chain theory as described by Tierney (1994), Tierney (1996) and for the basic definitions, the PhD dissertation of Johnson (2009) was also very useful.

To ease the notation we will always assume that the state space of the chain is $\mathbb{R}^k$, but the results remain valid in more general state spaces.

A.2 Basic definitions

Let $\Phi = \{\mathbf{X}^t : t \geq 0\}$ denote a discrete time Markov chain on $(\mathbb{R}^k, \mathscr{B}(\mathbb{R}^k))$. The construction of the chain begins from a starting value $\mathbf{X}^0$ which is specified by some initial distribution π_0. However, $\mathbf{X}^0$ is often set to some chosen value. From $\mathbf{X}^0$ the Markov chain evolves according to some *transition kernel* $\mathbb{P}$. We assume throughout that $\mathbb{P}$ has corresponding *transition density* k. The transition kernel is a function $\mathbb{P}(\mathbf{x}, A)$ such that for $t \geq 0$

$$\mathbb{P}(\mathbf{x}, A) = \mathbb{P}\left(\mathbf{X}^{t+1} \in A | \mathbf{X}^t = \mathbf{x}\right) = \int_A k(\mathbf{x}, \mathbf{z}) \mathbf{dz}. \tag{A.3}$$

for all $\mathbf{x} \in \mathbb{R}^k$ and $A \in \mathscr{B}(\mathbb{R}^k)$.

For any fixed $\mathbf{x}$, $\mathbb{P}(\mathbf{x}, \cdot)$ is a probability measure on $(\mathbb{R}^k, \mathscr{B}(\mathbb{R}^k))$. Hence, $\mathbb{P}$ is the distribution of the chain after one step given that it starts at $\mathbf{x}$. Different MCMC strategies such as the Gibbs or Metropolis-Hastings give rise to different transition kernels. The

chain constructed from π_0 and $\mathbb{P}$ is Markov since it satisfies the Markov property

$$\mathbb{P}(\mathbf{X}^{t+1}|\mathbf{X}^t, \mathbf{X}^{t-1}, \ldots, \mathbf{X}^0) = \mathbb{P}(\mathbf{X}^{t+1}|\mathbf{X}^t).$$

Further, the Markov chain is time invariant since $\mathbb{P}$ does not depend on the iteration number.

For a probability distribution ν on $(\mathbb{R}^k, \mathscr{B}(\mathbb{R}^k))$ with density $q(\mathbf{x})$ with respect to some measure μ i.e. for any set $A \in \mathscr{B}(\mathbb{R}^k)$, $\nu(A) = \int_A q(\mathbf{x})\mu(\mathbf{dx})$. A new probability distribution $\nu\,\mathbb{P}$ is defined by

$$\nu\,\mathbb{P}(A) = \int_{\mathbb{R}^k} \mathbb{P}(\mathbf{x}, A)q(\mathbf{x})\mu(\mathbf{dx}). \tag{A.4}$$

So if $\mathbf{X}^t$ has probability distribution ν, then $\mathbf{X}^{t+1}$ has probability distribution $\nu\,\mathbb{P}$. For ease of exposition we will often assume that μ is the Lebesgue measure. However, all methods herein extend beyond the Lebesgue setting.

Let $\mathbb{P}^t$ denote the t-step transition kernel corresponding to the t-step transition density k^t. Then we can write the t-step transition law of the chain as

$$\mathbb{P}^t(\mathbf{x}, A) = \mathbb{P}(\mathbf{X}^t \in A|\mathbf{X}^0 = \mathbf{x}) = \int_A k^t(\mathbf{x}, \mathbf{z})\mathbf{dx}, \tag{A.5}$$

where $A \in \mathscr{B}(\mathbb{R}^k)$ and k^t can be defined iteratively as

$$k^t(\mathbf{x}, \mathbf{z}) = \int_{\mathbb{R}^k} k(\mathbf{x}, \mathbf{v})k^{t-1}(\mathbf{v}, \mathbf{z})\mathbf{dv}.$$

If for some probability distribution π, with probability density $p(\cdot)$, we have that $\pi(A) = \pi\,\mathbb{P}(A)$ for all $A \in \mathscr{B}(\mathbb{R}^k)$ then π will be the invariant probability distribution of the Markov chain, i.e. $\mathbf{X}^t \sim \pi \Rightarrow \mathbf{X}^{t+1} \sim \pi = \pi\,\mathbb{P}$. Equivalently, π is invariant for the chain if

$$p(\mathbf{x}) = \int_{\mathbb{R}^k} k(\mathbf{z}, \mathbf{x})p(\mathbf{z})\mathbf{dz}$$

since for any $A \in \mathscr{B}(\mathbb{R}^k)$ integrating over both sides gives

$$\pi(A) = \int_A p(\mathbf{x})\mathbf{dx} = \int_A \left\{ \int_{\mathbb{R}^k} k(\mathbf{z}, \mathbf{x})p(\mathbf{z})\mathbf{dz} \right\} \mathbf{dx} = \int_{\mathbb{R}^k} \mathbb{P}(\mathbf{z}, A)p(\mathbf{z})\mathbf{dz} = \pi\,\mathbb{P}(A).$$

A useful result to prove that a chain has invariant probability distribution π is the *detailed balance condition*:

$$k(\mathbf{x}, \mathbf{z})p(\mathbf{x}) = k(\mathbf{z}, \mathbf{x})p(\mathbf{z}) \quad \text{for all } \mathbf{x}, \mathbf{z} \in \mathbb{R}^k. \tag{A.6}$$

If this property holds, the Markov chain is *reversible* with respect to π. To show that π is the invariant distribution we integrate over both sides of (A.6)

$$p(\mathbf{x}) = \int_{\mathbb{R}^k} k(\mathbf{x}, \mathbf{z})p(\mathbf{x})\mathbf{dz} = \int_{\mathbb{R}^k} k(\mathbf{z}, \mathbf{x})p(\mathbf{z})\mathbf{dz},$$

because $\displaystyle\int_{\mathbb{R}^k} k(\mathbf{x}, \mathbf{z})\mathbf{dz} = \mathbb{P}(\mathbf{x}, \mathbb{R}^k) = 1$.

Remark A.1. Reversibility is not required for π to be the invariant distribution of the Markov chain.

The total variation distance between two probability distributions v_1 and v_2 is defined as $||v_1 - v_2|| = \sup_{A \in \mathscr{B}(\mathbb{R})} |v_1(A) - v_2(A)|$.

A.3 Posterior distributions, ergodic Markov chains and convergence results

Suppose that we construct a Markov chain $\Phi = \{\theta^t : t \geq 0\}$ with transition kernel $\mathbb{P}$ that has invariant distribution

$$\pi(A|\mathbf{y}) = \int_A p(\theta|\mathbf{y})\mathbf{d}\theta \text{ with } A \in \mathscr{B}(\mathbb{R}^k). \tag{A.7}$$

Then (A.7) is the posterior distribution that is the main target of any Bayesian inference problem. If in this case, we want to use MCMC techniques to accurately estimate integrals such as (A.2) with Markov chain sample path averages such as

$$\bar{g}_N = \frac{1}{N+1} \sum_{t=0}^{N} g(\theta^t) \tag{A.8}$$

we need to ensure (A.8) converges to $\mathbb{E}(g(\theta)|\mathbf{y})$ from any initial value θ^0. A minimal requirement for this is *irreducibility*. Irreducibility guarantees that from any staring

point the chain is able to reach all the important parts of the state space (important with respect to some measure). Further, we need to ensure that there is no bias, i.e. the chain does not traverse the state space following a given pattern, such a chain is called *aperiodic*. For stronger convergence results we need an additional requirement, *Harris recurrence*. Harris recurrent means that for any starting value the chain visits all the important sets of the space state infinitely often. The formal definitions are

Definition A.3.1. *(ν-irreducible)* A Markov chain Φ is said to be ν-irreducible if there exists a non-zero σ-finite[1] measure ν on $(\mathbb{R}^k, \mathscr{B}(\mathbb{R}^k))$ such that for all $\theta \in \mathbb{R}^k$ and $A \in \mathscr{B}(\mathbb{R}^k)$ for which $\nu(A) > 0$, there exists some t such that $\mathbb{P}^t(\theta, A) > 0$. That is, Φ is ν-irreducible if every ν-positive set is accessible from any state $\theta \in \mathbb{R}^k$.

Definition A.3.2. *(Aperiodic)* A Markov chain Φ with invariant probability distribution $\pi(\cdot|\mathbf{y})$ is aperiodic if there do not exist $d \geq 2$ and disjoint subsets $\Theta_1, \Theta_2, \ldots, \Theta_d \in \mathscr{B}(\mathbb{R}^k)$ with $\mathbb{P}(\theta, \Theta_{i+1}) = 1$ for all $\theta \in \Theta_i$ $(1 \leq i \leq d-1)$, and $\mathbb{P}(\theta, \Theta_1) = 1$ for all $\theta \in \Theta_d$, such that $\pi(\Theta_1|\mathbf{y}) > 0$ (and hence $\pi(\Theta_i|\mathbf{y}) > 0$ for all i). That is, we cannot partition $\mathbb{R}^k$ such that Φ makes a regular tour through the partition.

Remark A.2. The chain is aperiodic, if there is positive probability that at time $t+1$ the chain remains in an arbitrary small neighborhood of θ^t.

Definition A.3.3. *(Harris Recurrent)* A Markov chain Φ is Harris recurrent if for any starting value $\theta \in \mathbb{R}^k$ the chain visits set $A \in \mathscr{B}(\mathbb{R}^k)$ infinitely often with probability one, i.e. $\mathbf{Pr}(\theta^t \in A \text{ i.o.}|\theta^0 = \theta) = 1$.

Definition A.3.4. *(Ergodic)* A Markov chain Φ is ergodic if it is ν-irreducible, aperiodic, Harris recurrent and has invariant probability distribution $\pi(\cdot|\mathbf{y})$.

The following Lemma is useful to show that a Markov chain is ergodic, see Lemma 1 from Tan and Hobert (2009).

[1] ν is σ-finite if there exist sets $A_1, A_2 \ldots$ such that $\cup_{i=1}^{\infty} A_i = \mathbb{R}^k$ and $\nu(A_i) > 0$ for all i

Lemma A.3.1. Suppose Φ is a Markov chain with transition kernel $\mathbb{P}$, transition density k and invariant probability measure $\pi(\cdot|\mathbf{y})$. If $k(\theta, \phi) > 0$ for all $\theta, \phi \in \mathbb{R}^k$, then Φ is ergodic.

We can now state the Theorem that will help us to estimate (A.2).

Theorem A.3.1 (**Ergodic Theorem**)**.** Suppose $\Phi = \{\theta^t : t \geq 0\}$ is an ergodic Markov chain on $(\mathbb{R}^k, \mathscr{B}(\mathbb{R}^k))$, with invariant probability distribution $\pi(\cdot|\mathbf{y})$. Also suppose that $\mathbb{E}(|g(\theta)||\mathbf{y}) < \infty$ for some $g : \mathbb{R}^k \to \mathbb{R}$. Then, for any starting value $\theta^0 = \theta$

$$\bar{g}_N \xrightarrow[N\to\infty]{\text{a.s.}} \mathbb{E}(g(\theta)|\mathbf{y}). \tag{A.9}$$

Theorem A.3.2. Suppose $\Phi = \{\theta^t : t \geq 0\}$ is an ergodic Markov chain on $(\mathbb{R}^k, \mathscr{B}(\mathbb{R}^k))$, with invariant probability distribution $\pi(\cdot|\mathbf{y})$. Then, for any starting value $\theta^0 = \theta$, the chain will converge to $\pi(\cdot|\mathbf{y})$ in total variation distance. That is

$$\lim_{t\to\infty} || \mathbb{P}^t(\theta, \cdot) - \pi(\cdot|\mathbf{y})|| = 0.$$

For convergence rates, see for example Tierney (1994).

Remark A.3. If the chain Φ is not Harris recurrent but is ν-irreducible and aperiodic with invariant probability distribution $\pi(\cdot|\mathbf{y})$, expression (A.9) holds for π-almost all $\theta^0 = \theta$ with probability one. That is, for π-almost all starting value $\theta^0 = \theta$

$$\mathbf{Pr}\left(\bar{g}_N \xrightarrow[N\to\infty]{} \mathbb{E}(g(\theta)|\mathbf{y})\right) = 1.$$

See Tierney (1994), Theorem 3, for the Ergodic Theorem and Roberts and Rosenthal (2004), Theorem 4, for the relaxed version.

Remark A.4. Marginalize out variables via MCMC ideas is straightforward, e.g. suppose that θ can be split into two components $(\theta_1, \theta_2) = \theta$ such that $(\theta_1, \theta_2) \in \mathbb{R}^{k_1} \times \mathbb{R}^{k_2} = \mathbb{R}^k$, hence if $\{(\theta_1^t, \theta_2^t) : t \geq 0\}$ is a Markov chain that satisfies the hypothesis of the Er-

godic Theorem

$$\begin{aligned} \mathbb{E}(g(\theta_2)|\mathbf{y}) &= \int_{\mathbb{R}^{k_2}} \int_{\mathbb{R}^{k_1}} g(\theta_2) p(\theta|\mathbf{y}) \mathbf{d}\theta_1 \mathbf{d}\theta_2 \\ &\approx \frac{1}{N+1} \sum_{t=0}^{N} g(\theta_2^t) \text{ for } N \text{ large enough.} \end{aligned}$$

A.4 MCMC algorithms

In this section we review well known MCMC methods to construct Markov chains with the correct invariant probability distribution: $\pi(\cdot|\mathbf{y})$. This is always the easiest part of any MCMC strategy, the hardest problem is to verify that the resulting chain is ergodic. In this case we will rely on Lemma A.3.1, however helpful advice to verify if a Markov chain is irreducible and aperiodic can be found in Tierney (1996), pp. 63 and 65.

A.4.1 The Gibbs sampler

The Gibbs sampler is a method of constructing a Markov chain $\{\theta^t : t \geq 0\}$ defined on $(\mathbb{R}^k, \mathscr{B}(\mathbb{R}^k))$ with invariant probability distribution (A.7) where it is not possible to sample from the joint density $p(\theta|\mathbf{y}) = p(\theta_1, \ldots, \theta_p|\mathbf{y})$ but it is possible to sample from its full conditional densities:

$$p(\theta_1|\theta_2, \ldots, \theta_p, \mathbf{y}), p(\theta_2|\theta_1, \theta_3, \ldots, \theta_p, \mathbf{y}), \ldots, p(\theta_p|\theta_1, \ldots, \theta_{p-1}, \mathbf{y}).$$

We can sample from the full joint conditionals via a deterministic sweep or alternatively via a random scan. The difference is that in the deterministic sweep the resulting chain is not reversible, see Robert and Casella (2004). In this Appendix we only review the Gibbs sampler generated via the deterministic sweep.

The Gibbs sampler algorithm is displayed in Algorithm (A.17).

Proposition A.4.1. If the density kernel k of the Gibbs sampler satisfies the conditions of Lemma A.3.1, Algorithm A.17 defines an ergodic Markov chain with invariant probability distribution (A.7).

Algorithm A.17 Gibbs sampler

Require: Given $(\theta_1^t, \ldots, \theta_p^t) = (\theta_1^t, \ldots, \theta_p^t)$, simulate $(\theta_1^{t+1}, \ldots, \theta_p^{t+1})$ via

1: $\theta_1^{t+1} \sim p(\theta_1^{t+1}|\theta_2^t, \theta_3^t \ldots, \theta_p^t, \mathbf{y})$.

2: $\theta_2^{t+1} \sim p(\theta_2^{t+1}|\theta_1^{t+1}, \theta_3^t, \ldots, \theta_p^t, \mathbf{y})$.

. . . .

p: $\theta_p^{t+1} \sim p(\theta_p^{t+1}|\theta_1^{t+1}, \theta_2^{t+1}, \ldots, \theta_{p-1}^{t+1}, \mathbf{y})$.

Proof. We will only work the case $p = 2$, the general case involves bulky manipulations with the conditional densities but the ideas are very similar.

The transition kernel of the chain is given by

$$\mathbb{P}\left((\theta_1^t, \theta_2^t), A\right) = \int_A p(\theta_1^{t+1}|\theta_2^t, \mathbf{y}) p(\theta_2^{t+1}|\theta_1^{t+1}, \mathbf{y}) \mathbf{d}\theta_1^{t+1} \mathbf{d}\theta_2^{t+1}. \tag{A.10}$$

So if $(\theta_1^t, \theta_2^t) \sim \pi$, then $(\theta_1^{t+1}, \theta_2^{t+1}) \sim \pi\,\mathbb{P}$ where from (A.4) is easy to see that

$$\begin{aligned}
\pi\,\mathbb{P}(A) &= \int_{\mathbb{R}^{k_2}} \int_{\mathbb{R}^{k_1}} \int_A p(\theta_1^t, \theta_2^t|\mathbf{y}) p(\theta_1^{t+1}|\theta_2^t, \mathbf{y}) p(\theta_2^{t+1}|\theta_1^{t+1}, \mathbf{y}) \mathbf{d}\theta_1^t \mathbf{d}\theta_2^t \mathbf{d}\theta_1^{t+1} \mathbf{d}\theta_2^{t+1}, \\
&= \int_A \int_{\mathbb{R}^{k_2}} p(\theta_2^t|\mathbf{y}) p(\theta_1^{t+1}|\theta_2^t, \mathbf{y}) p(\theta_2^{t+1}|\theta_1^{t+1}, \mathbf{y}) \mathbf{d}\theta_2^t \mathbf{d}\theta_1^{t+1} \mathbf{d}\theta_2^{t+1}, \\
&= \int_A p(\theta_1^{t+1}, \theta_2^{t+1}|\mathbf{y}) \mathbf{d}\theta_1^{t+1} \mathbf{d}\theta_2^{t+1}, \\
&= \pi(A|\mathbf{y}).
\end{aligned}$$

Hence if $k(\theta^t, \theta^{t+1}) = p(\theta_1^{t+1}|\theta_2^t, \mathbf{y}) p(\theta_2^{t+1}|\theta_1^{t+1}, \mathbf{y}) > 0$, Lemma A.3.1 guarantees the ergodicity of the chain. □

Remark A.5. If the full conditionals used to generate the Gibbs sampler are positive, from Lemma A.3.1, the resulting Markov chain is ergodic.

A.4.2 The Metropolis-Hastings

The Metropolis-Hastings is a method of constructing a Markov chain $\{\theta^t : t \geq 0\}$ defined on $(\mathbb{R}^k, \mathscr{B}(\mathbb{R}^k))$ with invariant probability distribution (A.7). This algorithm is based on proposing values sampled from an instrumental distribution, $q(\phi|\theta)$, which are then accepted with a certain probability that reflects how likely it is that they are from

the target posterior density $p(\theta|\mathbf{y})$. The Metropolis-Hastings algorithm is displayed in Algorithm A.18.

Algorithm A.18 Metropolis-Hastings

Require: Given $\theta = \theta^t$, simulate θ^{t+1} via

1: Draw $\phi^* \sim q(\phi \mid \theta^t)$.

2: Take

$$\theta^{t+1} = \begin{cases} \phi^* & \text{with probability } \alpha(\theta^t, \phi^*), \\ \theta^t & \text{with probability } 1 - \alpha(\theta^t, \phi^*). \end{cases}$$

where

$$\alpha(\theta, \phi) = \begin{cases} \min\left\{1, \dfrac{p(\phi|\mathbf{y})q(\theta|\phi)}{p(\theta|\mathbf{y})q(\phi|\theta)}\right\} & \text{if } p(\theta|\mathbf{y})q(\phi|\theta) > 0, \\ 1 & \text{otherwise.} \end{cases}$$

Remark A.6. In Algorithm A.18, the target density, $p(\theta|\mathbf{y})$, should be known only up to a proportionality constant.

Proposition A.4.2. Algorithm A.18 defines a Markov chain with invariant probability distribution (A.7).

Proof. In this case we can use the detailed balance condition to prove that $\pi(\cdot|\mathbf{y})$ is the invariant distribution of the chain. The transition density of the chain is given by

$$k(\theta^t, \phi^*) = \alpha(\theta^t, \phi^*)q(\phi^*|\theta^t) + \left\{1 - r(\theta^t)\right\}\delta_{\theta^t}(\phi^*),$$

where $r(\theta) = \displaystyle\int_{\mathbb{R}^k} \alpha(\theta, \phi)q(\phi|\theta)d\phi$.

The reversibility of the Metropolis-Hastings is easily established by noting

$$\begin{aligned} \alpha(\theta^t, \phi^*)q(\phi^*|\theta^t)p(\theta^t|\mathbf{y}) &= \min\left\{1, \frac{p(\phi^*|\mathbf{y})\ q(\theta^t|\phi^*)}{p(\theta^t|\mathbf{y})\ q(\phi^*|\theta^t)}\right\} q(\phi^*|\theta^t)p(\theta^t|\mathbf{y}) \\ &= \alpha(\phi^*, \theta^t)q(\theta^t|\phi^*)p(\phi^*|\mathbf{y}) \quad \text{and} \\ \left\{1 - r(\theta^t)\right\}\delta_{\theta^t}(\phi^*)p(\theta^t|\mathbf{y}) &= \left\{1 - r(\phi^*)\right\}\delta_{\phi^*}(\theta^t)p(\phi^*|\mathbf{y}) \end{aligned}$$

these two conditions show that $k(\theta^t, \phi^*)p(\theta|\mathbf{y}) = k(\phi^*, \theta^t)p(\phi^*|\mathbf{y})$. □

We can use again Lemma A.3.1 to prove the ergodicity of the chain, this will clearly depend on the instrumental distribution $q(\phi|\theta)$ and on $p(\theta|\mathbf{y})$.

To finish this section we have to mention that the Gibbs sampler can be obtained as a particular case of the Metropolis-Hastings where the proposals are always accepted with probability one, see Robert and Casella (2004), Chapter 10.

A.4.3 The slice sampler

The slice sampler is a method of constructing a Markov chain $\{\theta^t : t \geq 0\}$ defined on $(\mathbb{R}^k, \mathscr{B}(\mathbb{R}^k))$ with invariant probability distribution (A.7), where after the introduction of strategic latent variables we can derive a Gibbs sampler with easy to sample full joint conditionals. The idea is as follows, consider the density given by $p(\theta|\mathbf{y}) \propto p(\theta)l(\mathbf{y}|\theta)$, where l is a non-negative invertible function on θ (not necessarily a density) and $p(\theta)$ a density. Now, suppose that it is not possible to sample directly from $p(\theta|\mathbf{y})$. Hence, we introduce a latent variable u such that $p(\theta, u|\mathbf{y}) \propto p(\theta)\,\mathbb{1}\{u < l(\mathbf{y}|\theta)\}$. It is clear that integrating out u we go back to the original model. The full conditionals for the Gibbs sampler will be given by

$$\begin{aligned} p(u|\theta, \mathbf{y}) &\propto \mathrm{U}(0, l(\mathbf{y}|\theta)), \\ p(\theta|u, \mathbf{y}) &\propto p(\theta)\,\mathbb{1}\{u < l(\mathbf{y}|\theta)\}. \end{aligned}$$

Since l, as a function of θ, has an inverse it is possible to find the set $A_u = \{\theta|l(\mathbf{y}|\theta) > u\}$, thus to sample from $p(\theta|u, \mathbf{y})$ we will sample from a truncated version of $p(\theta)$.

Remark A.7. The decomposition $p(\theta|\mathbf{y}) \propto p(\theta)l(\mathbf{y}|\theta)$ is not unique, and this fact can be used when constructing the joint density containing the latent variable.

The main aim under slice sampler is to avoid the use of a rejection sampler, such as the Metropolis-Hastings, and derive an efficient and easy to code MCMC strategy.

Remark A.8. This idea is extended to the case where

$$p(\theta|\mathbf{y}) \propto p(\theta)\prod_{i=1}^{n} l_i(\mathbf{y}|\theta).$$

Here we will need to introduce the latent variables $u_1, \ldots, u_n$, and work with the set $A_u = \{\theta | l_i(\mathbf{y}|\theta) > u_i, \text{ for } i = 1, \ldots, n\}$.

For more on slice sampler techniques see Damien, Wakefield, and Walker (1999) for the main ideas, and Damien and Walker (2001) for ways to sample from truncated distributions.

A.4.4 Hybrid strategies

We have described how to use the Gibbs, Metropolis-Hasting and slice sampler in pure form to generate a Markov chain with invariant probability distribution (A.7). But they can be combined into hybrid strategies, these are commonly called cycles and mixtures, see Tierney (1994). In a cycle there are transition kernels $\mathbb{P}_1, \ldots, \mathbb{P}_k$: each kernel is used in turn and when the last one is used, the cycle is restarted. As an example suppose θ can be split into (θ_1, θ_2) and that we can sample from $\theta_1|\theta_2$ directly, as in the Gibbs sampler, but direct sampling from $\theta_2|\theta_1$ is not possible, thus to sample from $\theta_2|\theta_1$ we use a Metropolis-Hastings or a slice sampler. Is easy to see that the resulting chain will have invariant probability distribution $\pi(\cdot|\mathbf{y})$. In a mixture there are transition kernels $\mathbb{P}_1, \ldots, \mathbb{P}_k$ and positive probabilities $p_1, \ldots, p_k$ ($p_j > 0$ and $\sum_j p_j = 1$). At each step one of the kernels is selected according with these probabilities. Here if the invariant probability distribution for each kernel is $\pi(\cdot|\mathbf{y})$, the invariant probability distribution for the mixture $\mathbb{Q} = p_1 \mathbb{P}_1 + \cdots + p_k \mathbb{P}_k$ will be $\pi(\cdot|\mathbf{y})$, this is straightforward since

$$\pi\mathbb{Q} = (p_1 \pi \mathbb{P}_1 + \cdots + p_k \pi \mathbb{P}_k) = \pi.$$

A.5 Practical considerations: starting values, burn in period and diagnosing MCMC convergence

The distribution of the chain depends on starting values θ^0 hence to initiate any MCMC algorithm we would like to chose $\theta^0 \sim \pi(\cdot|\mathbf{y})$. The problem is that this is rarely possible,

thus whether we find ways to chose θ^0 close to the mode of the posterior distribution or we simply sample the starting values form the prior. In either case, to obtain accurate estimates, we need to ensure that the sample path of the chain used to calculate $\bar{g}_N$ is free from the influence of θ^0. A common practice is to discard the first iterations of the simulation, hence effectively reducing the dependence of $\bar{g}_N$ on θ^0. The discarded iterations are usually called *burn-in* period, see Gilks, Richardson, and Spiegelhalter (1996). Now, if the Markov chain is able to move quickly to and through the support of the posterior distribution, sometimes referred to as *good mixing*, for small burn-in period $\bar{g}_N$ will be almost independent of θ^0. In contrast, if the chain gets stuck in small regions of the state space for long periods of time, longer burn-in periods will be needed to reduce the dependence between $\bar{g}_N$ and θ^0.

The mixing behavior of the chain also determines the rate of convergence of the estimator $\bar{g}_N$ to $\mathbb{E}(g(\theta)|\mathbf{y})$, and so determines the length of the chain N required for accurate inference, see the Ergodic Theorem, expression (A.9). Convergence is faster if the chain exhibits good mixing behavior. Therefore we need to asses the mixing behavior of the chain, this is usually called *diagnosing MCMC convergence*. In the literature there are several proposals to diagnose MCMC converge, select N and the length of the burn-in period, see the R package `coda` (Plummer, Best, Cowles, and Vines (2006)) and references therein. We have taken a less formal approach, assessing the convergence of our chains on the basis of graphical output of the results, and compare results from parallel chains run from different starting values: trace of the chain, plots of ergodic averages, density estimates etc. With this we expect to detect: movement away from the starting points, influence of the starting values of the different chains and in general asses if the chains have achieved stable behavior.

Bibliography

Antoniak, C. E. (1974): "Mixtures of Dirichlet Processes with Applications to Bayesian Nonparametric Problems," *The Annals of Statistics*, 2(6), 1152–1174.

Becker, M. (2012): "PearsonDS: Pearson Distribution System," R package version 0.93, http://CRAN.R-project.org/package=PearsonDS.

Blackwell, D. (1973): "Discreteness of Ferguson Selections," *Annals of Statistics*, 1(2), 356–358.

Blackwell, D., and J. B. MacQueen (1973): "Ferguson distributions via Pólya urn schemes," *Annals of Statistics*, 1(2), 353–355.

Brunner, L. J., and A. Y. Lo (1989): "Bayes Methods for a Symmetric Unimodal Density and its Mode," *The Annals of Statistics*, 17(4), 1550–1566.

Burkard, R. E., M. Dell'Amico, and S. Martello (2009): *Assignment problems*. SIAM, Philadelphia.

Byron, H. (2012): "LaplacesDemon: Software for Bayesian Inference," R package version 12.05.07, http://cran.r-project.org/web/packages/LaplacesDemon/index.html.

Cappé, O., C. P. Robert, and T. Rydén (2003): "Reversible jump, birth-and-death and more general continuous time Markov chain Monte Carlo samplers," *Journal of the Royal Statistical Society, Series B*, 65(3), 679–700.

Carlin, B. P., and S. Chib (1995): "Bayesian Model Choice via Markov Chain Monte Carlo Methods," *Journal of the Royal Statistical Society, Series B*, 57(3), 473–484.

Carlin, B. P., and T. A. Louis (2000): *Bayes and Empirical Bayes Methods for Data Analysis*, Texts in statistical science. Taylor & Francis, second edn.

Casella, G., C. P. Robert, and M. T. Wells (2004): "Mixture models, latent variables and partitioned importance sampling," *Statistical Methodology*, 1(1-2), 1–18.

Celeux, G. (1998): "Bayesian Inference for Mixtures: The Label Switching Problem," in *COMPSTAT 98*, ed. by R. Payne, and P. J. Green, pp. 227–232. Physica-Verlag, Berlin.

Celeux, G., M. Hurn, and C. P. Robert (2000): "Computational and Inferential Difficulties with Mixture Posterior Distributions," *Journal of the American Statistical Association*, 95(451), 957–970.

Chopin, N., T. Lelièvre, and G. Stoltz (2012): "Free energy methods for Bayesian inference: efficient exploration of univariate Gaussian mixture posteriors," *Statistics and Computing*, 22, 897–916.

Damien, P., J. Wakefield, and S. G. Walker (1999): "Gibbs Sampling for Bayesian Non-Conjugate and Hierarchical Models by Using Auxiliary Variables," *Journal of the Royal Statistical Society. Series B*, 61(2), 331–344.

Damien, P., and S. G. Walker (2001): "Sampling Truncated Normal, Beta and Gamma Densities," *Journal of Computational and Graphical Statistics*, 10(2), 206–215.

Dellaportas, P., and I. Papageorgiou (2006): "Multivariate mixtures of normals with unknown number of components," *Statistics and Computing*, 16, 57–68.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977): "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society. Series B*, 39(1), 1–38.

Devroye, L. (1997): "Random variate generation for multivariate unimodal densities," *Journal ACM Transactions on Modeling and Computer Simulation*, 7(4), 447–477.

Diebolt, J., and C. P. Robert (1994): "Estimation of Finite Mixtures Distributions through Bayesian Sampling," *Journal of the Royal Statistical Society, Series B*, 56(2), 363–375.

Escobar, M. D. (1988): "Estimating the Means of Several Normal Populations by Nonparametric Estimation of the Distribution of the Means," Unpublished ph.d. thesis, Department of Statistics, Yale University.

——— (1994): "Estimating Normal Means with a Dirichlet Process Prior," *Journal of the American Statistical Association*, 89(425), 268–277.

Escobar, M. D., and M. West (1995): "Bayesian Density Estimation and Inference Using Mixtures," *Journal of the American Statistical Association*, 90(430), 577–588.

Feller, W. (1971): *An Introduction to Probability Theory and Its Applications*. John Wiley & Sons, New York, pp. 157-158.

Ferguson, T. S. (1973): "A Bayesian Analysis of Some Nonparametric Problems," *The Annals of Statistics*, 1(2), 209–230.

——— (1983): "Bayesian Density Estimation by Mixtures of Normal Distributions," in *Recent advances in statistics: papers in honor of Herman Chernoff on his sixtieth birthday*, ed. by H. Chernoff, J. S. Rustagi, M. H. Rizvi, and D. Siegmund, pp. 287–302. Academic Press, New York.

Fernandez, C., and M. F. J. Steel (1998): "On Bayesian Modeling of Fat Tails and Skewness," *Journal of the American Statistical Association*, 93(441), 359–371.

Frühwirth-Schnatter, S. (2001): "Markov Chain Monte Carlo Estimation of Classical and Dynamic Switching and Mixture Models," *Journal of the American Statistical Association*, 96(453), 194–209.

——— (2006): *Finite Mixture and Markov Switching Models*. Springer, New York.

Galassi, M. et al. (2009): "GNU Scientific Library Reference Manual," Manual, http://www.gnu.org/software/gsl/.

Gelfand, A. E., and A. F. M. Smith (1990): "Sampling-Based Approaches to Calculating Marginal Densities," *Journal of the American Statistical Association*, 85(410), 398–409.

Geweke, J. (2007): "Interpretation and Inference in Mixture Models: Simple MCMC Works," *Computational Statistics & Data Analysis*, 51(7), 3529–3550.

Ghosal, S. (2010): "Dirichlet Process, Related Priors and Posterior Asymptotics," in *Bayesian Nonparametrics*, ed. by L. N. Hjort, C. C. Holmes, P. Müller, and S. G. Walker, pp. 35–79. Cambridge University Press, Cambridge, U.K.

Gilks, W. R., S. Richardson, and D. J. Spiegelhalter (1996): "Introducing Markov Chain Monte Carlo," in *Markov Chain Monte Carlo in Practice*, ed. by W. Gilks, S. Richardson, and D. Spiegelhalter, pp. 1–19, London. Chpaman & Hall.

Godsill, S. J. (2001): "On the Relationship Between Markov Chain Monte Carlo Methods for Model Uncertainty," *Journal of Computational Graphical Statistics*, 10(2), 230–240.

Green, P. J. (1995): "Reversible Jump Markov Chain Monte Carlo Computation and Bayesian Model Determination," *Biometrika*, 82(4), 711–732.

——— (2003): "Trans-Dimensional Markov Chain Monte Carlo," in *Highly Structured Stochastic Systems*, ed. by P. J. Green, N. L. Hjort, and S. Richardson, pp. 179–198. Oxford University Press, Oxford, U.K.

Griffin, J., and C. C. Holmes (2010): "Computational Issues Arising in Bayesian Nonparametric Hierarchical Models," in *Bayesian Nonparametrics*, ed. by L. N. Hjort, C. C. Holmes, P. Müller, and S. G. Walker, pp. 208–222. Cambridge University Press, Cambridge, U.K.

Grün, B., and F. Leisch (2009): "Dealing with Label Switching in Mixture Models Under Genuine Multimodality," *Journal of Multivariate Analysis*, 100(5), 851–861.

Hornik, K. (2012): "clue: Cluster ensembles," R package version 0.3-44, http://CRAN.R-project.org/package=clue.

Ishwaran, H., and L. F. James (2001): "Gibbs Sampling Methods for Stick-Breaking Priors," *Journal of the American Statistical Association*, 96(453), 161–173.

Jaeger, A. (2005): "Large File Support in Linux," Web page, http://www.suse.de/~aj/linux_lfs.html.

Jasra, A., C. C. Holmes, and D. A. Stephens (2005): "Markov Chain Monte Carlo Methods and the Label Switching Problem in Bayesian Mixture Modeling," *Statistical Science*, 20(1), 50–67.

Johnson, A. A. (2009): "Geometric Ergodicity of Gibbs samplers," Ph.d thesis, University of Minnesota, http://www.macalester.edu/~ajohns24/Thesis.pdf.

Kalli, M., J. E. Griffin, and S. G. Walker (2011): "Slice Sampling Mixture Models," *Statistics and Computing*, 21(1), 93–105.

Khinchin, A. Y. (1938): "On unimodal distributions (in Russian)," *Trams. Res. Inst. Math. Mech. (University of Tomsk)*, 2(2), 1–7.

Komárek, A. (2009): "A new R package for Bayesian estimation of multivariate normal mixtures allowing for selection of number of components and interval-censored data," *Computational Statistics and Data Analysis*, 53(12), 3932–3947.

Kottas, A., and A. E. Gelfand (2001): "Bayesian Semiparametric Median Regression Modeling," *Journal of the American Statistical Association*, 96(456), 1456–1468.

Lijoi, A., and I. Prünster (2010): "Models Beyond the Dirichlet Process," in *Bayesian Nonparametrics*, ed. by L. N. Hjort, C. C. Holmes, P. Müller, and S. G. Walker, pp. 80–136. Cambridge University Press, Cambridge, U.K.

Lo, A. Y. (1984): "On a Class of Bayesian Nonparametric Estimates: I. Density Estimates," *The Annals of Statistics*, 12(1), 351–357.

MacEachern, S. N. (1994): "Estimating normal means with a conjugate style dirichlet process prior," *Communications in Statistics - Simulation and Computation*, 23(3), 727–741.

Mächler, M. (2011): "nor1mix: Normal (1-d) Mixture Models (S3 Classes and Methods)," R package version 1.1-3, http://CRAN.R-project.org/package=nor1mix.

Martin, J. M., K. Mengersen, and C. P. . Robert (2005): "Bayesian Modellinig and Inference on Mixture of Distributions," in *Handbook of Statistics 25*, ed. by D. Dey, and C. R. Rao, pp. 459–507. Elsevier, Holland.

Neal, R. M. (2000): "Markov Chain Sampling Methods for Dirichlet Process Mixture Models," *Journal of Computational and Graphical Statistics*, 9(2), 249–265.

Nobile, A., and A. T. Fearnside (2007): "Bayesian Finite Mixtures with an Unknown Number of Components: The Allocation Sampler," *Statistics and Computing*, 17(2), 147–162.

Papaspiliopoulos, O., and G. O. Roberts (2008): "Retrospective Markov Chain Monte Carlo methods for Dirichlet Process Hierarchical Models," *Biometrika*, 95(1), 169–186.

Papastamoulis, P., and G. Iliopoulos (2010): "An Artificial Allocations Based Solution to the Label Switching Problem in Bayesian Analysis of Mixtures of Distributions," *Journal of Computational and Graphical Statistics*, 19(2), 313–331.

——— (2011): "On the Convergence Rate of Random Permutation Sampler and ECR Algorithm in Missing Data Models," *Methodology and Computing in Applied Probability*, pp. 1–12.

Pearson, K. (1894): "Contributions to the Mathematical Theory of Evolution," *Philosophical Transactions of the Royal Society Series A*, 185, 71–110.

Peng, R. D., and J. Leeuw (2002): "An Introduction to the .C Interface to R," Manual, UCLA: Academic Technology Services, Statistical Consulting Group, http://www.ats.ucla.edu/stat/r/library/interface.pdf.

Pitman, J. (2006): "Combinatorial Stochastic Processes," Lecture notes, Springer-Verlag, Berlin, http://works.bepress.com/jim_pitman/1.

Plummer, M., N. Best, K. Cowles, and K. Vines (2006): "CODA: Convergence Diagnosis and Output Analysis for MCMC," *R News*, 6(1), 7–11, http://CRAN.R-project.org/doc/Rnews/Rnews_2006-1.pdf.

Postman, M., J. P. Huchra, and M. J. Geller (1986): "Probes of large-scale structures in the Corona Borealis region," *Astrophysical Journal*, 92, 1238–1247.

Quintana, F. A., M. F. J. Steel, and J. T. A. S. Ferreira (2009): "Flexible Univariate Continuous Distributions," *Bayesian Analysis*, 4(3), 497–522.

R Core Team (2012): "R: A Language and Environment for Statistical Computing," Manual, R Foundation for Statistical Computing, Vienna, Austria, http://www.R-project.org/.

Richardson, S., and P. J. Green (1997): "On Bayesian Analysis of Mixtures with an Unknown Number of Components (with discussion)," *Journal of the Royal Statistical Society, Series B*, 59(4), 731–792.

——— (1998): "Corrigendum: On Bayesian Analysis of Mixtures with an Unknown Number of Components," *Journal of the Royal Statistical Society, Series B*, 60(3), 661.

Rigby, R. A., and D. M. Stasinopoulos (2005): "Generalized Additive Models for Location, Scale and Shape," *Journal of the Royal Statistical Society, Series C*, 54(3), 507–554.

Ripley, B. (1987): *Stochastic Simulation*, Wiley Series in Probability and Statistics. J. Wiley.

Robert, C. P. (1996): "Mixtures of Distributions: Inference and Estimation," in *Markov Chain Monte Carlo in Practice*, ed. by W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, pp. 441–464, New York.

Robert, C. P., and G. Casella (2004): *Monte Carlo Statistical Methods*. Springer-Verlag, New York.

Roberts, G. O., and J. S. Rosenthal (2004): "General state space Markov chains and MCMC algorithms," *Probability Surveys*, 1, 20–71.

Rodríguez, C. E., and S. G. Walker (2012): "Univariate Bayesian Nonparametric Mixture Modeling with Unimodal Kernels," *To appear in Statistics and Computing*.

——— (2013): "Label Switching in Bayesian Mixture Models: deterministic relabeling strategies," *To appear in Journal of Computational and Graphical Statistics*.

Rudin, W. (1976): *Principles of Mathematical Analysis*. McGraw-Hill, New York.

Sethuraman, J. (1994): "A Constructive Definition of Dirichlet Priors," *Statistica Sinica*, 4, 639–650.

Sisson, S. A. (2005): "Transdimensional Markov Chains: A Decade of Progress and Future Perspectives," *Journal of the American Statistical Association*, 100(471), 1077–1089.

Smith, A. F. M., and G. O. Roberts (1993): "Bayesian Computations via the Gibbs Sampler and Related Markov Chain Monte Carlo Methods," *Journal of the Royal Statistical Society, Series B*, 55(4), 3–23.

Stephens, M. (1997): "Bayesian Methods for Mixtures of Normal Distributions," unpublished ph.d thesis, University of Oxford, Oxford, http://stephenslab.uchicago.edu/MSpapers/DPhilMS.ps.gz.

——— (2000a): "Bayesian Analysis of Mixture Models with an Unknown Number of Components - An Alternative to Reversible Jump Methods," *The Annals of Statistics*, 28(1), 40–74.

——— (2000b): "Dealing with Label Switching in Mixture Models," *Journal of the Royal Statistical Society, Series B*, 62(4), 795–809.

Tan, A., and J. P. Hobert (2009): "Block Gibbs Sampling for Bayesian Random Effects Models With Improper Priors: Convergence and Regeneration," *Journal of Computational and Graphical Statistics*, 18(4), 861–878.

Tao, D. (1989): "On multivariate unimodal distributions," Msc. thesis, University of British Columbia, https://circle.ubc.ca/handle/2429/27411.

Tierney, L. (1994): "Markov Chains for Exploring Posterior Distributions (with discussion)," *The Annals of Statistics*, 22(4), 1701–1762.

——— (1996): "Introduction to the General State-Space Markov Theory," in *Markov Chain Monte Carlo in Practice*, ed. by W. Gilks, S. Richardson, and D. Spiegelhalter, pp. 59–74, London. Chpaman & Hall.

Viallefont, V., S. Richardson, and P. J. Green (2002): "Bayesian Analysis of Poisson Mixtures," *Journal of Nonparametric Statistics*, 14(1-2), 181–202.

Walker, S. G. (2007): "Sampling the Dirichlet Mixture Model with Slices," *Communications in Statistics*, 36(1), 45–54.

Walker, S. G., P. Damien, P. W. Laud, and A. F. M. Smith (1999): "Bayesian Nonparametric Inference for Random Distributions and Related Functions (with discussion)," *Journal of the Royal Statistical Society, Series B*, 61(3), 485–527.

Wiper, M., D. Rios-Insua, and F. Ruggeri (2001): "Mixtures of Gamma Distributions with Applications," *Journal of Computational & Graphical Statistics*, 10(3), 440–454.