Bayesian analysis for mixtures of discrete distributions with a non-parametric component

Baba Bukar Alhaji Bukar

A Thesis Submitted for the Degree of

Doctor of Philosophy

Department of Mathematical Sciences

University of Essex

May 2016

Dedicated to

My kids, source of my happiness

Ahmad, Abba and Ummi

ABSTRACT

Bayesian finite mixture modelling is a flexible parametric modelling approach for classification and density fitting. Many application areas require distinguishing a *signal* from a *noise* component. In practice, it is often difficult to justify a specific distribution for the *signal* component, therefore the *signal* distribution is usually further modelled via a mixture of distributions. However, modelling the *signal* as a mixture of distributions is computationally challenging due to the difficulties in justifying the exact number of components to be used and due to the label-switching problem. The use of a non-parametric distribution to model the *signal* component is proposed. This new methodology leads to more accurate parameter estimation, smaller classification error rate and smaller false non-discovery rate in the case of discrete data. Moreover, it does not incur the label-switching problem. An application of the method to data generated by ChIP-sequencing experiments is shown.

A one-dimensional Markov random field model is proposed, which accounts for the spatial dependencies in the data. The methodology is also applied to ChIP-seq data, which shows that the new method detected more genes enriched regions than similar existing methods at the same false discovery rate.

PUBLICATIONS

[1] Alhaji, B. B., Dai, H., Hayashi, Y., Vinciotti, V., Harrison, A., & Lausen, B. (2015). Bayesian analysis for mixtures of discrete distributions with a non-parametric component. *Journal of Applied Statistics*, 1-17 http://www.tandfonline.com/doi/abs/10.1080/02664763.2015.1100594.

[2] Alhaji, B.B., Dai, H., Hayashi, Y., Vinciotti, V., Harrison, A., & Lausen, B. (2015). Analysis of chip-seq data via bayesian finite mixture models with a non-parametric component. In: Wilhelm, A., Kestler, H. A. (eds.), Analysis of Large and Complex Data, European Conference on Data Analysis, Bremen, July, 2014, Series: Studies in Classification, Data Analysis, and Knowledge Organization. http://www.springer.com/gb/book/9783319252247.

DECLARATION

The work in this thesis is based on research carried out at the Statistics and Actuarial Science Group, Department of Mathematical Sciences, University of Essex, United Kingdom. No part of this thesis has been submitted elsewhere for any other degree or qualification, and it is all my own work, unless referenced, to the contrary, in the text.

ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to my supervisor Dr Hongsheng Dai for his continuous support during my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge. One could not simply wish for a better or friendlier supervisor.

My deepest appreciation goes to my co-supervisors, Prof Berthold Lausen and Dr Andrew Harrison, for their encouragement, insightful comments, and expertise that greatly assisted this research.

I would also like to take this opportunity to express my heartfelt gratitude to my supervisory board member, Prof Peter M Higgins, and all staff in the Department of Mathematical Sciences at University of Essex for their great help and support. I am also deeply indebted to my wife, who supported me throughout this venture.

I am very much obliged to the Nigerian Defence Academy for nominating me for sponsorship under the Tertiary Education Trust Fund (TETFund) scholarship scheme.

Finally, I place on record my sense of gratitude to one and all who, directly or indirectly, have lent their hand in this venture.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

In the late twentieth century, Bayesian methods began to be recognized as a key to major changes in statistics. The Bayesian statistics approach has a lot of advantages over the frequentist approach. The frequentist approach is based on repeated samples from a particular model. In the Bayesian statistics approach, however, the user can formally incorporate prior information into the model. Introducing a prior brings extra information into the model, and results in posterior estimates that combine two sources of information. These sources of information are the previous belief we had about the process (prior) and the information we already have in the data (likelihood). Introducing a prior may improve on frequentist estimators in terms of precision. Prior probabilities in Bayesian methods are subjective. Although frequentists view this as a drawback, advocates of the Bayesian approach opine that the subjectivity is inevitable, and they further argue that the frequentist approach also involves subjective choices. Several researchers adopt the Bayesian method due to its flexibility and consistency in the face of uncertainties. The Bayesian method provides an atmosphere which is conducive to structuring the data and knowledge about the data in order to yield conclusive solutions to problems.

Before 1990s, the major challenge faced by the Bayesian method was the difficulty in evaluating the normalizing constant as a result of computational intractability. Direct computation of the posterior distribution is intractable numerically or analytically, owing to the dimensionality and the complexity of the model. But when new computational methods were developed, and powerful computer programs became available, the Bayesian method was enthusiastically embraced. After the discovery of posterior simulation techniques called Markov Chain Monte Carlo (MCMC) methods, Bayesian methods made a significant impact in statistical theory, and provided solutions to pressing questions in many application areas (Robert & Casella 2009). MCMC is a technique of constructing a Markov process such that the distribution of the samples after a certain number of steps, called the stationary distribution, is approximately the required posterior distribution (Gelman et al. 2014, Gilks et al. 1996). The use of MCMC algorithms has the promise of computing the posterior distribution, and evaluating the posterior estimates.

1.1 Finite Mixture models

The Bayesian approach to mixture models has attracted great interest among researchers, as a tool for parameter estimation and density fittings. Before the discovery of MCMC, mixture models provided solutions to a few specialized cases (Jasra et al. 2005). Estimation of parameters via Bayesian analysis of mixture models when the number of components is assumed known became routine after the paper of Diebolt & Robert (1994). Bayesian analysis of mixture models for an unknown number of components is now possible using the methodologies of reversible jump MCMC (Richardson & Green 1997), Birth and Death

MCMC (Nobile & Fearnside 2007, Stephens 2000*a*) and mixture of Dirichlet processes (Antoniak 1974, Escobar & West 1995). The basic idea behind the mixture model is that we think of observations which originate from different underlying sources with different underlying distributions (Pearson 1894). From a statistical point of view we refer to the mixture model as a probabilistic model that tells us the presence of some underlying sub-populations in an overall population. The finite mixture model represents heterogeneity in a finite number of latent classes. It enables us to model data with different distributions and account for the underlying heterogeneity. It enables us to fit the data, use the Bayesian method to estimate the parameters for the separate distributions, and obtain membership probabilities of each component for each observation. Applications of the finite mixture model include machine learning such as clustering, latent class models, pattern recognition, computer vision and survival analysis.

In the Bayesian paradigm, the mixture model is feasible with the advent of MCMC simulation. Although the existing literature has shown that the finite mixture model can be inferred in a simple and effective way in a Bayesian estimation framework, attention is mostly focused on parametric mixture modelling, when the mixture components are having the same type of distributions. For example, all the component distributions could be Poisson with different means or all the component distributions could be negative Binomial (NB) with different parameters (even though, in practice, it is not necessary that all the distributions will be of the same kind). Modelling such a situation causes a persistent challenge in the diagnostic of MCMC convergence due to the following reasons.

The first reason is the challenge of non-identifiability of the components parameters, the so-called *label-switching problem*. During the MCMC simulation, the components' weight

and parameters interchange making it difficult to assess if the chain has reached a stationary distribution.

Another challenge to the Bayesian mixture model is the task of selecting the number of components for the mixtures. There is a trade-off among the users of the mixture model in the model order selection. There is fear of over-fitting the data when too many components are used in the mixture. On the other hand, flexibility of the mixture model will be lost with too few components in the mixture. These, and other related challenges, are our motivation for the use of the mixture model with a non-parametric component.

1.2 Markov random field model

A Hidden Markov Model (HMM) is described as a stochastic model governed by a Markov process that has a finite number of states, and a set of random functions related to each state. Rabiner (1989) used HMM for speech recognition, and the latter is continuously being used in numerous applications, such as modelling economic and financial data, biological sequence analysis and in other areas of artificial intelligence and pattern recognition.

HMMs have received widespread attention recently as they provide a handy extension of independent mixture models to allow for dependent data. The independent assumption in mixture models is removed by considering successive correlation of the data through the component from which they are generated.

A one-dimensional Markov Random Field (MRF) model is any one-dimensional Markov chain. It is, therefore, a first order Markov chain, which satisfies the Markov condition, $h(z_i = m|z_{-i}) = h(z_i = m|z_{i-1}, z_{i+1})$, where $z_{-i} = \{z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n\}$. MRF models are

applied in many research areas, such as computer vision, machine learning and biological sequence analysis.

1.3 Research objectives

The overarching aim of the research here is to investigate good models for the analysis of discrete data sets that involve classification into two groups, consisting of signal and noise, using the Bayesian method. Two types of mixture models are focused in this thesis, and these are the finite mixture model and one-dimensional Markov random field mixture model. Specifically, the thesis addresses both methodological and applied aspects of Bayesian modelling and is designed to achieve the following objectives:

- to demonstrate that mixture models are suitable and flexible semi-parametric frameworks for estimation and classification;
- to develop a finite mixture model with a non-parametric component for the analysis of discrete distributions, when the interest is to distinguish signal from noise for independent observations;
- to develop a Markov random field model with a non-parametric component as an extension to the finite mixture model, to account for spatial dependencies in discrete observations.

1.4 Source of data

To achieve these research objectives, ChIP-Seq data sets generated by Ramos et al. (2010) for identifying the genomic regions bound by the histone acetyltransferases are used. For each region in the genome, the data report the number of bound fragments that align to that region. Some regions contain large tags (signal) and the rest consist of fewer tags (noise). A higher value suggests that the corresponding region is most likely to be bound by the protein of interest. Therefore, the underlying data consist of sub-populations, which can be inferred by a two-component mixture model with signal and noise components.

1.5 Structure of the thesis

The rest of this work is further divided into six chapters and three appendices.

Chapter 2 focuses on Bayesian analysis for mixture models. The concepts of Bayesian statistical inference, including prior distribution and MCMC methods, are explained. A general overview of mixture models, including motivations and challenges of mixture models are provided. The finite mixture model is introduced in this Chapter. The Chapter further discusses the hidden Markov model and the Markov random field model.

Chapter 3 introduces ChIP-seq data. The experimental work flow for ChIP-seq and description of the ChIP-seq data sets used in the thesis are discussed in the Chapter. Discussions on the parametric distributions used for modelling the noise component are also provided. These parametric distributions are: Poisson, negative Binonial, zero-inflated Poisson, and zero-inflated negative Binomial distributions.

Chapters 4, 5 and 6 consist of the original work. The Bayesian mixture model for discrete

distributions with parametric and non-parametric components is proposed in Chapter 4. Extensive simulation studies under different scenarios are carried out. The results obtained are compared with fully parametric mixture models. Chapter 5 provides application of the proposed method to ChIP-sequence data sets in order to detect enriched gene regions along the genome and the results are compared with existing methods. Chapter 6 is an extension of the proposed method to account for spatial dependencies by incorporating the Markov property into the new model. The proposed one-dimensional MRF model is discussed in this Chapter. Simulation studies and ChIP-seq data analysis are also carried out. The result of the data analysis is compared with existing methods.

To conclude this thesis, Chapter 7 summarizes and discusses the main contributions of the research and provides possible suggestion for future research work.

Appendix A provides prior sensitivity analysis and data analysis trace plots for the proposed method, Appendix B provides simulation plots for the one-dimensional Markov random field model, and Appendix C provides the R codes implemented in the thesis for simulations and data analyses.

CHAPTER 2

BAYESIAN ANALYSIS FOR MIXTURE MODELS

2.1 The statistical model and Bayesian inference

Specification of a statistical model is the cornerstone of all statistical inference. A statistical model is a collection of probability distribution functions that explain the generation of observed data (Williams 2008). Probability distribution functions fall into two categories: non-parametric and parametric distributions. A non-parametric distribution is a distribution that does not rely on assumptions about the shape or form of the probability distribution from which the data are drawn (Sheskin 2003). In contrast, a distribution is said to be a parametric if it relies on assumptions about the shape of the probability distribution, and can be described based on a finite number of parameters of the assumed distribution. For example, we denote the parametric statistical model for an observation x_i as

$$x_i \sim h(x_i|\boldsymbol{\theta}), \tag{2.1}$$

where $h(x_i|\boldsymbol{\theta})$ is the probability density (or mass) function of x_i, viewed as a probability of realizing x_i as a function of some parameters $\boldsymbol{\theta}$. The estimation process is to choose that value of $\boldsymbol{\theta}$ that would maximize the probability that we would actually observe x_i. In other words, we find the parameter values $\boldsymbol{\theta}$ that maximize the following function:

$$l(\boldsymbol{\theta}|x_i) = h(x_i|\boldsymbol{\theta}).$$

Here, $l(\boldsymbol{\theta}|x_i)$ is known as the likelihood function. Thus, the likelihood function is viewed as a function of the unknown parameter $\boldsymbol{\theta}$, which indexes the distribution from which x_i is generated (Press 2009). For example, suppose x_i is a Bernoulli random variable (binary 0,1 values), then the likelihood based on a single observation is

$$l(\lambda|x_i) = \lambda^{x_i}(1-\lambda)^{1-x_i}. \tag{2.2}$$

Furthermore, if $x_1, \ldots, x_n$ are independent observations drawn from some parametric distribution, then the complete likelihood for $\boldsymbol{\theta}$ based on n observations is given by

$$l(\boldsymbol{\theta}|\boldsymbol{x}) \propto \prod_{i=1}^{n} h(x_i|\boldsymbol{\theta}). \tag{2.3}$$

If we still assume that sample of $x_1, \ldots, x_n$ is independently drawn from a Bernoulli distribution, then the likelihood function is

$$\begin{aligned} l(\lambda|x_1, \ldots, x_n) &= h(x_1, \ldots, x_n|\lambda) \\ &\propto \prod_{i=1}^{n} \lambda^{x_i}(1-\lambda)^{1-x_i} = \lambda^{\sum_{i=1}^{n} x_i}(1-\lambda)^{n-\sum_{i=1}^{n} x_i}. \end{aligned} \tag{2.4}$$

The likelihood function plays an important role in statistical inference, especially as a method of estimating parameter $\boldsymbol{\theta}$. By likelihood we mean how likely the parameter $\boldsymbol{\theta}$ (selected model) agrees with the observed data. In the classical maximum likelihood estimation (MLE) method, we find the derivative of the log likelihood function,

$$log(l(\boldsymbol{\theta})) = \sum_{i=1}^{n} log(h(x_i|\boldsymbol{\theta})), \tag{2.5}$$

and set its derivative equal to zero

$$\frac{\partial}{\partial \boldsymbol{\theta}} log(l(\boldsymbol{\theta})) = 0. \tag{2.6}$$

In Bayesian inference, the parameter $\boldsymbol{\theta}$ is assumed to be a random variable. Let $g(\boldsymbol{\theta})$ be the probability distribution of the parameter $\boldsymbol{\theta}$, referred to as the prior distribution. The prior distribution allows us to reflect our opinion (if any) concerning the circumstance before the data is observed. The inference concerning $\boldsymbol{\theta}$ is then obtained by Bayes' theorem.

The posterior distribution of $\boldsymbol{\theta}$ is given by

$$\begin{aligned} f(\boldsymbol{\theta}|\boldsymbol{x}) &= \frac{l(\boldsymbol{\theta}|\boldsymbol{x})g(\boldsymbol{\theta})}{\int l(\boldsymbol{\theta}|\boldsymbol{x})g(\boldsymbol{\theta})d\boldsymbol{\theta}} \\ &\propto l(\boldsymbol{\theta}|\boldsymbol{x})g(\boldsymbol{\theta}). \end{aligned} \tag{2.7}$$

The quantity $c = \int l(\boldsymbol{\theta}|\boldsymbol{x})g(\boldsymbol{\theta})d\boldsymbol{\theta}$ is the normalizing constant for the function $\boldsymbol{\theta} \mapsto l(\boldsymbol{\theta}|\boldsymbol{x})g(\boldsymbol{\theta})$ (Chen et al. 2012).

Adopting Bayesian analysis, therefore, can provide inference about the parameter $\boldsymbol{\theta}$ conditioned on the data. It provides the flexibility of incorporating external information as prior belief about the parameters.

2.2 The prior distribution

In Bayesian analysis, before the observations are taken into account, the uncertainty of the parameter is expressed as a probability distribution. The prior probability distribution (simply the prior distribution) of the parameter $\boldsymbol{\theta}$ is a key part of modelling uncertainty in parameter $\boldsymbol{\theta}$. The prior distributions can be classified as belonging to an informative or a non-informative prior. A prior is informative when the current information is combined with information gathered from past experience, such as a previous study or expert opinion (Bernardo & Smith 2009). Informative prior distributions are proper prior (they integrate or sum to 1). However, informative prior distributions are more subjective; that is, they may be only meaningful to the particular analyst that used them (Press 2009).

Conversely, when we do not have prior knowledge or have little prior information

about the unknown parameters before any data are taken, a non-informative or a flat prior is preferred. The non-informative prior distributions are more preferred by statisticians, because they are more objective than the informative prior distributions. On the other hand, a non-informative prior usually leads to an improper posterior (nonintegrable posterior density) (Gelman 2002). In the non-informative prior setting, the prior $g(\boldsymbol{\theta})$ may be considered to be a uniform distribution, that is, all possible outcome of the parameter $\boldsymbol{\theta}$ have the same probability. As such, a non-informative prior has little impact on the inference (relative to the information in the likelihood) (Marin & Robert 2014).

Another form of prior probability distribution is the conjugate prior. A prior distribution is said to be a conjugate if the prior probability distribution $g(\boldsymbol{\theta})$ and the posterior distribution $f(\boldsymbol{\theta}|\boldsymbol{x})$ are of the same parametric family of distributions, and the prior is said to be a conjugate prior for the likelihood function. For example, if the likelihood is Poisson, $x \sim Pois(\lambda)$, a conjugate prior on λ is Gamma(r, v) distribution. Consequently, the posterior distribution of λ is also Gamma distribution. In order to distinguish them from the model parameters, the prior distribution parameters are referred to as hyperparameters. For example, if we denote the parameters for the likelihood as $\boldsymbol{\theta}$, and assume that the parameters for the prior distribution are known and are denoted as $\boldsymbol{\eta}$, thus the prior can be written as $g(\boldsymbol{\theta}) = g(\boldsymbol{\theta}|\boldsymbol{\eta})$, where $\boldsymbol{\eta}$ are hyperparameters (Carlin & Louis 2011). Nevertheless, the power of Bayesian methods largely depends on the proper use of a prior distribution.

In simplest Bayesian models, when the posterior distribution is analytically tractable to a constant, the unknown parameters can easily be simulated from the posterior distribution if the posterior is from a recognizable distribution. However, this may not be possible always, especially for complex models, or when a non-conjugate prior is used. In the

next section we discuss an alternative approach to generating samples from the posterior distribution, when the normalizing constant is analytically intractable. The approach is called Markov chain Monte Carlo methods.

2.3 Markov chain Monte Carlo method

In Bayesian analysis, MCMC methods allow one to generate samples from (2.7), when the normalizing constant is analytically intractable owing to the dimensionality of the model parameters, and expectation of quantities of interest are approximated from these samples (Press 2009). Basically, MCMC algorithm constructs a Markov chain such that after a certain number of steps, the chain converges to a stationary distribution. The stationary distribution is the desired posterior distribution. The simulated values from the MCMC methods are clearly dependent samples by its very nature (Carlin & Louis 2011). In practice, the initial draws are typically discarded to allow the effect of the starting value to wear off. This is referred to as the burn-in period (Press 2009). We briefly describe two most commonly used MCMC methods: the Gibbs sampler and Metropolis-Hastings algorithm. Robert & Casella (2009) provides a detailed discussion on MCMC theory and implementation.

2.3.1 The Gibbs sampler

Gibbs sampling (Geman & Geman 1984, Turchin 1971) is a technique for generating random variables from a conditional distribution. A sample is drawn from the distribution of each parameter in turn, given the current values of the other parameters (Casella & George 1992).

The distribution from which the sample is drawn is called the full conditional distribution. The sampler is efficient when the parameters have full conditional distributions that are easy to sample from.

Suppose $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K)$ is the blocks of the parameter vector. The full posterior distribution $f(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K|\boldsymbol{x})$, as described in Bolstad (2011) is given by

$$f(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K|\boldsymbol{x}) \propto l(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K|\boldsymbol{x})g(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K), \tag{2.8}$$

where $l(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K|\boldsymbol{x})$ is the likelihood and $g(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K)$ is the prior distribution. And the full conditional distribution for parameters $\boldsymbol{\theta}_k$ has the general form

$$f(\boldsymbol{\theta}_k|\boldsymbol{\theta}_{-k}, \boldsymbol{x}) = f(\boldsymbol{\theta}_k|\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{k-1}, \boldsymbol{\theta}_{k+1}, \ldots, \boldsymbol{\theta}_K, \boldsymbol{x}), \tag{2.9}$$

where $\boldsymbol{\theta}_{-k}$ is the set of all the other parameters not in block k. In the procedure of Gibbs sampling, we set the initial estimates for each parameter $\boldsymbol{\theta}_k$, and then $\boldsymbol{\theta}_k$ is drawn from its full conditional distribution by a cyclical sampling scheme through the parameter blocks in turn given the most recent estimates of the other parameter blocks and the data. The

procedure is summarized as follows (see Bolstad (2011)).

Algorithm 1: The Gibbs sampler

1. Set $t = 0$ and start from an arbitrary point $\boldsymbol{\theta}^{(0)} = (\theta_1^{(0)}, \ldots, \theta_K^{(0)})$

2. For $t = 1, 2, \ldots,$ until convergence; $k = 1, \ldots, K$, simulate $\theta_k^{(t)}$ from

   - simulate $\theta_1^{(t)} \sim f(\theta_1 | \theta_2^{(t-1)}, \ldots, \theta_K^{(t-1)}, \boldsymbol{x})$
   - simulate $\theta_2^{(t)} \sim f(\theta_2 | \theta_1^{(t)}, \theta_3^{(t-1)}, \ldots, \theta_K^{(t-1)}, \boldsymbol{x})$
   -
   -
   - simulate $\theta_K^{(t)} \sim f(\theta_K | \theta_1^{(t)}, \ldots, \theta_{K-1}^{(t)}, \boldsymbol{x})$.

3. The stationary distribution $\boldsymbol{\theta}^{(N)} = (\theta_1^{(N)}, \ldots, \theta_K^{(N)})$ is the true posterior distribution of $f(\theta_1, \ldots, \theta_K | \boldsymbol{x})$.

Algorithm 1 gives a Markov chain of $(\theta_1^{(t)}, \ldots, \theta_K^{(t)})$ $(t = 1, 2, \ldots,)$. It is our intention to give only a brief outline of the Gibbs sampling scheme. Detailed modifications to this scheme can be obtained in Casella & George (1992) and Gelman et al. (2014). We have implemented Gibbs sampler in the following Chapters and the sampling schemes are briefly outlined.

2.3.2 The Metropolis-Hastings algorithm

When the prior $g(\boldsymbol{\theta})$ and the likelihood $l(\boldsymbol{\theta}|\boldsymbol{x})$ are not a conjugate pair, then their full conditionals may not be available in close form. Alternative to Gibbs sampling method, the Metropolis-Hastings (MH) algorithm is used (Carlin & Louis 2011). This is another power-

ful procedure that produces a correlated sequence of samples from the target distribution that may be difficult to make draws by classical independence methods. The Metropolis-Hastings algorithm is an extension of Metropolis algorithm (Metropolis et al. 1953), where a parameter value is drawn from its posterior distribution by proposing a new parameter value given the current value of the parameter, and the draw is either accepted or rejected according to a specified probability.

Metropolis-Hastings (MH) is a rejection algorithm which proposes a new parameter value $\boldsymbol{\theta}^*$ from a specified proposal distribution $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(i-1)})$ that is, in practice, easy to simulate. Then the process accepts or rejects the candidate based on the acceptance ratio. The generic MH algorithm is provided in Algorithm 2.

Algorithm 2: Metropolis-Hastings algorithm

Initialization: $\boldsymbol{\theta}^{(0)} \sim q(\boldsymbol{\theta})$;
for $i = 1$ *to* N **do**
  Propose: $\boldsymbol{\theta}^* \sim q(\boldsymbol{\theta}^{(i)}|\boldsymbol{\theta}^{(i-1)})$;
  Acceptance Probability

$$r = \alpha(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(i-1)}) = \min\left\{1, \frac{f(\boldsymbol{\theta}^*|\boldsymbol{x})q(\boldsymbol{\theta}^{(i-1)}|\boldsymbol{\theta}^*)}{f(\boldsymbol{\theta}^{(i-1)}|\boldsymbol{x})q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(i-1)})}\right\}$$

  $u \sim \text{Uniform}(u; 0, 1)$;
  if $u < r$ **then**
    Accept the proposal: $\boldsymbol{\theta}^{(i)} \leftarrow \boldsymbol{\theta}^*$
  else
    Reject the proposal: $\boldsymbol{\theta}^{(i)} \leftarrow \boldsymbol{\theta}^{(i-1)}$
  end
end

In MH algorithm applications, one important issue is the choice of the proposal density $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(i-1)})$. The choice of proposal density is crucial for rapid convergence of the algorithm. A proposal density can be a symmetric. A proposal density is symmetric if $q(\boldsymbol{\theta}^{(i)}|\boldsymbol{\theta}^{(i-1)}) = q(\boldsymbol{\theta}^{(i-1)}|\boldsymbol{\theta}^{(i)})$. Gaussian distributions and Uniform distributions can be straight-

forward choices of a symmetric proposals centered at the current state of the chain. For example, for a Normal proposal we have that $\boldsymbol{\theta}^* = \boldsymbol{\theta}^{(i-1)} + Normal(0, \sigma)$, such that pdf for $Normal(\boldsymbol{\theta}^* - \boldsymbol{\theta}^{(i-1)}; 0, \sigma) = Normal(\boldsymbol{\theta}^{(i-1)} - \boldsymbol{\theta}^*; 0, \sigma)$ (Yildirim 2012). This procedure is called the Random-walk Metropolis algorithm (Chib & Greenberg 1995, Yildirim 2012).

Another idea for the choice of a proposal density is an independent proposal. Here, the candidate $\boldsymbol{\theta}^*$ is drawn independently from the proposal distribution. That is the proposal $q(\boldsymbol{\theta}^*|\boldsymbol{\theta}^{(i-1)}) = q(\boldsymbol{\theta}^*)$ does not depend on $\boldsymbol{\theta}^{(i-1)}$ (Gilks et al. 1996). The Gibbs sampling is considered a special case of the Metropolis-Hastings algorithm where specifically, a proposal from the full conditional distribution has a Metropolis-Hastings ratio of 1 - implying that the proposal is always accepted (Gelman et al. 2014).

Though presented in isolation, the Metropolis-Hastings and Gibbs sampling procedures can be implemented in a single algorithm - called Metropolis-within-Gibbs algorithm. For each draws of $\boldsymbol{\theta}_k^{(t)}$, as in regular Gibbs sampling, either a full conditional distribution or a Metropolis draw is used. This is particularly useful if some of the full conditionals have a known form, but some of them do not (Geweke et al. 2003). For comprehensive discussions and implementation with examples on Metropolis-Hastings algorithm refer to Chib & Greenberg (1995).

In summary MCMC method is one of the standard tools of statisticians' apparatus in the Bayesian model inference because it is efficient, fast and easily implemented.

2.4 Mixture model

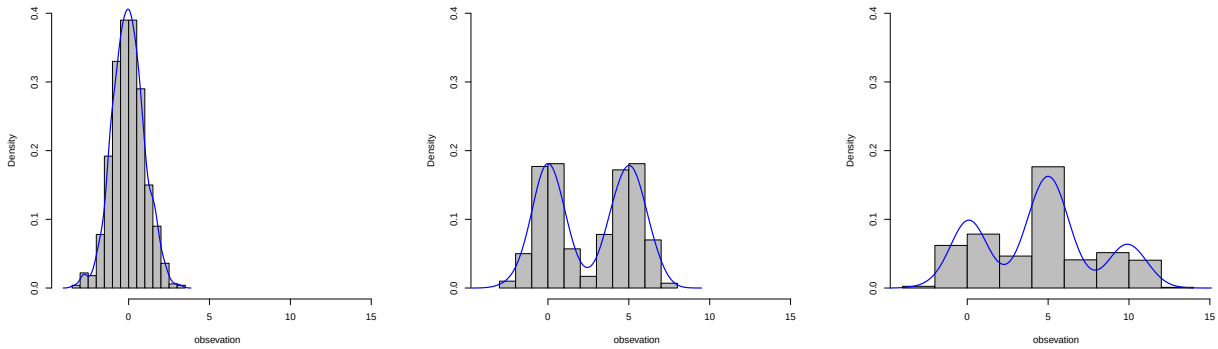
This section introduces the concept of mixture model, and explains the motivation for its use in this thesis, and as well as how it is used in a complex statistical problems.

2.4.1 Motivation

In certain situations the observed data may be so complex such that, use of a single parametric distribution may be insufficient for making inference. The structure of the data may consist of several homogeneous subgroups. One may be interested in identifying these subgroups, which may provide useful information for inference. Mixture models provide a solution to this problem by assuming the observations are drawn from some parametric distributions.

Consider the simulated data on Figure 2.1 for illustration purpose. From Figure 2.1(a), it is reasonable to use a single parametric distribution to model such data, because it is likely drawn from a single known parametric distribution (say normal distribution). Whereas in Figures 2.1(b) and 2.1(c), the observed data comes from more than one distribution. By fitting a single parametric distribution, one may not address the multimodality expressed in the data. Therefore, we fit a mixture model through appropriate choice of components to achieve true representation of the data. It should be noted that, apart from multimodality in the observed data, mixture model is also useful in modelling heavy-tailed densities and skewed densities (Venturini et al. 2008). Mixture model can provide a useful statistical tool that is suited for the analysis of many different data types and distributions (McLachlan & Peel 2004). These distributions may come from any parametric family, continuous or

discrete, univariate or multivariate. A mixture model is capable of approximating any arbitrary distribution.



(a) (b) (c)

Figure 2.1: Histogram of simulated data to illustrate the use of mixture models for (a) one Normal distribution, (b) two Normal distributions and (c) three Normal distributions.

Bayesian approach to fitting mixture model involves specifying prior distributions over the parameters of the mixture model. That is allowing probability statements to express uncertainty involved about the unknown parameters. Bayesian mixture of distributions have gained popularity in a range of application areas, such as computer science, astronomy, economics, engineering, robotics, ecology, and as demonstrated in this thesis, genetics. The following section reviews two selections of mixture model used in this thesis. These models are finite mixture and hidden Markov mixture models.

2.5 Finite mixture model

In the Bayesian modelling framework it is possible to handle mixtures with infinite number of components. This results in a mixture model with an infinite number of latent classes. Alternatively, the choice of number of components is treated as fixed (Diebolt & Robert

1994). In this case, one is not faced with the issue of estimating the latent classes. Finite mixture model involved modelling with categorical latent variables that express heterogeneity in a finite number of latent classes (McLachlan & Peel 2004). Detailed treatment of finite mixture model theory and implementation is provided, for example, in McLachlan & Peel (2004), Marin et al. (2005) and Frühwirth-Schnatter (2006).

2.5.1 Model specification and priors

Let $x_1, \ldots, x_n$ be a sample of size n which is assumed to be drawn from a mixture of distributions

$$h(\boldsymbol{x}|\boldsymbol{\pi}, \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k h_k(x_i|\boldsymbol{\theta}_k), \tag{2.10}$$

where K is the number of components, $\boldsymbol{\theta}_k$ is the component parameter, $h_k(x_i|\boldsymbol{\theta}_k)$ is the density of the component and π_k is the component weight, such that

$$\pi_k \geq 0, \quad \text{and} \quad \sum_{k=1}^{K} \pi_k = 1.$$

The likelihood is given by

$$l(\boldsymbol{\pi}, \boldsymbol{\theta}|\boldsymbol{x}) = \prod_{i=1}^{n} \left[\sum_{k=1}^{K} \pi_k h_k(x_i|\boldsymbol{\theta}_k) \right]. \tag{2.11}$$

Mixture models can be interpreted in terms of missing or incomplete data (Diebolt & Robert 1994). Let $z_i, \quad (i = 1, \ldots, n)$ be an indicator variable, indicating to which component observation x_i belongs. Therefore, $\boldsymbol{z} = (z_1, \ldots, z_n)$ are random variables which are independent

and identically distributed with probability mass function given as

$$\Pr(z_i = k|\boldsymbol{\pi}, \boldsymbol{\theta}) = \pi_k, \quad (i = 1, \ldots, n, \ k = 1, \ldots, K).$$

Assume that the observations $x_1, \ldots, x_n$ are drawn independently from

$$h(x_i|z_i = k, \boldsymbol{\pi}, \boldsymbol{\theta}) = h(x_i|\boldsymbol{\theta}_k), \quad (i = 1, \ldots, n),$$

then the complete likelihood function conditioned on the data and the latent variable (Frühwirth-Schnatter 2006) is

$$\begin{aligned} l(\boldsymbol{\pi}, \boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{z}) &= \prod_{i=1}^{n} \pi_{z_i} h(x_i|\boldsymbol{\theta}_{z_i}) \\ &= \prod_{k=1}^{K} \left[\prod_{i:z_i=k} \pi_k h(x_i|\boldsymbol{\theta}_k) \right] \\ &= \prod_{k=1}^{K} \left[\pi_k^{\sum_{i=1}^{n} I(z_i=k)} \prod_{i:z_i=k} h(x_i|\boldsymbol{\theta}_k) \right]. \end{aligned} \tag{2.12}$$

Denote the prior on the component parameters, which depends on the distribution family underlying the mixture distribution (Frühwirth-Schnatter 2006) as

$$\boldsymbol{\theta} \sim g(\boldsymbol{\theta}). \tag{2.13}$$

The prior distribution for the weights for K components is usually assumed to be Dirichlet distribution

$$(\pi_1, \ldots, \pi_K) \sim Dir(\eta_1, \ldots, \eta_K), \tag{2.14}$$

which is defined by the hyperparameters, $\eta_1, \ldots, \eta_K$. A prior assumed on $\boldsymbol{\pi}$ is non-informative if $\eta_1 =, \ldots, = \eta_K = 1$. For a two-component mixture, the Dirichlet distribution reduces to a Beta distribution.

2.5.2 Posterior and model inference

Throughout this thesis we assume that the number of components K, is known. Statistical inference for mixtures with an unknown number of components is beyond the scope of this thesis.

McLachlan & Peel (2004) and Frühwirth-Schnatter (2006) provide a comprehensive review on classical and Bayesian mixture models inference. The goal of the model inference is to infer the unknown component parameters $(\boldsymbol{\pi}, \boldsymbol{\theta})$ given the component indicators $\boldsymbol{z}$ and the observations $\boldsymbol{x}$. Therefore, Bayesian mixture model for sampling from the complete data posterior is proportional to

$$f(\boldsymbol{\pi}, \boldsymbol{\theta}|\boldsymbol{z}, \boldsymbol{x}) \propto l(\boldsymbol{\pi}, \boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{z}) g(\boldsymbol{\pi}) g(\boldsymbol{\theta}). \tag{2.15}$$

where $l(\boldsymbol{\pi}, \boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{z})$ is the complete data likelihood in (2.12), and $g(\boldsymbol{\pi})$ and $g(\boldsymbol{\theta})$ are the prior distributions for the unknown parameters.

MCMC can be used to draw realizations independently from the posterior in (2.15) using Gibbs sampling scheme by breaking into the following full conditionals

$$\boldsymbol{z} \sim f(\boldsymbol{z}|\boldsymbol{\pi}, \boldsymbol{\theta}, \boldsymbol{x}); \tag{2.16}$$

$$\boldsymbol{\theta} \sim f(\boldsymbol{\theta}|z, \boldsymbol{x}, \pi); \tag{2.17}$$

$$\boldsymbol{\pi} \sim f(\boldsymbol{\pi}|z, \boldsymbol{x}, \boldsymbol{\theta}). \tag{2.18}$$

Starting with the latent variable, the full conditional for the latent variable given the component parameters and the observations is

$$f(\boldsymbol{z}|\boldsymbol{\pi}, \boldsymbol{\theta}, \boldsymbol{x}) \propto h(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{\pi}, \boldsymbol{\theta}). \tag{2.19}$$

Based on the full posterior distribution in (2.19), the posterior probability of $z_i = k$ given x_i, such that the i^{th} observation belonging to k^{th} component, can be computed (Frühwirth-Schnatter 2006, McLachlan & Peel 2004)

$$\begin{aligned} \Pr(z_i = k|x_i, \boldsymbol{\pi}, \boldsymbol{\theta}) &= \frac{\pi_k h(x_i|\boldsymbol{\theta}_k)}{\sum_{l=1}^{K} \pi_l h(x_i|\boldsymbol{\theta}_l)} \\ &\propto \pi_k h(x_i|\boldsymbol{\theta}_k). \end{aligned} \tag{2.20}$$

Therefore, the posterior of z_i follows a Multinomial,

$$z_i \sim Mult(\Pr(z_i = 1|x_i, \boldsymbol{\pi}, \boldsymbol{\theta}), \ldots, \Pr(z_i = K|x_i, \boldsymbol{\pi}, \boldsymbol{\theta})). \tag{2.21}$$

Conditional on $\boldsymbol{z}$ and $\boldsymbol{x}$, $\boldsymbol{\theta}$ is sampled from the posterior (Frühwirth-Schnatter 2006)

$$f(\boldsymbol{\theta}_k|\boldsymbol{x}, \boldsymbol{z} = k) \propto \left[\prod_{i:z_i=k} h(x_i|\boldsymbol{\theta}_k)\right] g(\boldsymbol{\theta}_k). \tag{2.22}$$

If $g(\boldsymbol{\theta}_k)$ is a conjugate, then it will be of recognizable parametric form. And given $\boldsymbol{z}$, $\boldsymbol{\pi}$ is sampled from the posterior distribution (Frühwirth-Schnatter 2006)

$$f(\boldsymbol{\pi}|\boldsymbol{z}) \propto \prod_{k=1}^{K} \pi_k{}^{\sum_{i=1}^{n} I(z_i=k)} g(\pi). \tag{2.23}$$

For a conjugate prior on $\boldsymbol{\pi}$, the full conditional for $\boldsymbol{\pi}$ follows a Dirichlet distribution given by

$$\boldsymbol{\pi} \sim Dir(n_1 + \eta_1, \ldots, n_K + \eta_K), \tag{2.24}$$

where $n_K = \sum_i (z_i = k),\ k = 1, \ldots, K$.

Given these full conditionals, Gibbs sampling for finite mixture model can be carried out, described as follows. Sample draws are made from the distributions of each parameter in turn, given the current values of the other parameters (Casella & George 1992). For example, set an initial value for $\boldsymbol{\theta}$ as $\boldsymbol{\theta}^{(0)}$. Given $\boldsymbol{x}$ and $\boldsymbol{\theta}^{(0)}$, then $\boldsymbol{z}$ depends on $\boldsymbol{x}$ and $\boldsymbol{\theta}^{(0)}$. Then $\boldsymbol{z}^{(0)}$ is sampled from the conditional distribution of $\boldsymbol{z}$ given $\boldsymbol{x}$ and $\boldsymbol{\theta}^{(0)}$. The distribution of $\boldsymbol{\theta}$ depends on $\boldsymbol{x}$ and $\boldsymbol{z}$. Then a new value $\boldsymbol{\theta}^{(1)}$ is sampled given $\boldsymbol{z}^{(0)}$ and $\boldsymbol{x}$. Repeating the process T times produces T steps Markov chain that has a stationary distribution as its posterior. A Gibbs sampler for mixture model is summarized in Algorithm 3.

Algorithm 3: The Gibbs sampler for mixture model

- **Initialization:** Set $\boldsymbol{\pi}^{(0)}$, $\boldsymbol{z}^{(0)}$ and $\boldsymbol{\theta}^{(0)}$ arbitrarily
- **Step t.** For $t = 1, 2, \ldots$
  1. update $z_i^{(t)}$ $(i = 1, \ldots, n)$ from,
  $$\Pr(z_i^{(t)} = k | \pi_k^{(t-1)}, \boldsymbol{\theta}_k^{(t-1)}, x_i) \propto \pi_k^{(t-1)} h(x_i | \boldsymbol{\theta}_k^{(t-1)})$$
  2. update $\boldsymbol{\pi}^{(t)}$ from $f(\boldsymbol{\pi}|\boldsymbol{z}^{(t)})$;
  3. update $\boldsymbol{\theta}^{(t)}$ from $f(\boldsymbol{\theta}|\boldsymbol{z}^{(t)}, \boldsymbol{x})$

2.6 Challenge of mixture model

The mixture model in (2.10) is said to be a symmetric mixture model if the parameters $\boldsymbol{\theta}_k$s have the same dimension and given a common parameter $\boldsymbol{\theta}_0$, $h_1(x, \boldsymbol{\theta}_0) = \cdots = h_K(x, \boldsymbol{\theta}_0)$ for any value of x. Otherwise the mixture model is said to be asymmetric. For example, the mixture density, $h(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{\pi}) = \sum_{k=1}^{K} \pi_k N(x; \mu_k, \sigma_k)$, is said to be symmetric since each normal component density have mean μ_k and variance $1/\sigma_k$. Estimating the marginal density for such symmetric mixture model in a Bayesian analysis from the MCMC draws may lead to a poor result. The major challenge of such an approach is the difficulty in assessing the convergence of the MCMC samplers due to an identifiability problem (i.e., the label switching problem).

2.6.1 The label switching problem

The label switching problem occurs when the likelihood is invariant under permutation of the component parameters, and there are $K!$ permutations. So, the posterior will inherit the invariance of the likelihood, because the priors are symmetric. Consequently, in any

MCMC algorithm, the component label permutes multiple times between iterations of the sampler. As a result of this, the posterior estimate of the characteristic of the component becomes useless (Rodríguez & Walker 2014). In summary, label switching problem is caused by the symmetry in the likelihood of the model parameters (Stephens 2000*b*).

To illustrate this, let V_K be the set of the permutations of the labels $\{1, \ldots, K\}$. If for some $v \in V_K$, we have that $v(\boldsymbol{\pi}, \boldsymbol{\theta}) := ((\pi_{v(1)}, \ldots, \pi_{v(K)}), (\boldsymbol{\theta}_{v(1)}, \ldots, \boldsymbol{\theta}_{v(K)}))$. Then

$$\begin{aligned} h(\boldsymbol{x}|\boldsymbol{\pi}, \boldsymbol{\theta}) &= \pi_1 h(x; \boldsymbol{\theta}_1) + \cdots + \pi_K h(x; \boldsymbol{\theta}_K) \\ &= \pi_{v(1)} h(x; \boldsymbol{\theta}_{v(1)}) + \cdots + \pi_{v(K)} h(x; \boldsymbol{\theta}_{v(K)}) \\ &= h(\boldsymbol{x}|v(\boldsymbol{\pi}, \boldsymbol{\theta})). \end{aligned} \tag{2.25}$$

Then under this, the likelihood function for n observations is:

$$l(\boldsymbol{\theta}; \boldsymbol{x}) = \prod_{i=1}^{n} h(x_i|\boldsymbol{\pi}, \boldsymbol{\theta}) = \prod_{i=1}^{n} h(x_i|v(\boldsymbol{\pi}, \boldsymbol{\theta})) = l(v(\boldsymbol{\pi}, \boldsymbol{\theta}); x).$$

That is the likelihood function is invariant with respect to the permutation of the component parameters. The prior distribution is the same for all the permutation of the component parameters, if there is no prior information that distinguishes between the mixture components. Hence, under this symmetric priors across components, the posterior will also be exchangeable (invariant with respect to the permutation of the component parameters). As such the sampler will encounter a symmetries of the posterior distribution during MCMC simulation (Xie & Chen 2012). This make the ergodic averages to estimate characteristics of the component meaningless.

To solve this problem, the initial attempt was to add an identifiability constraint on the parameters through the prior distribution. This is to obtain a unique labelling by breaking the symmetry in the posterior distribution. For this approach see Diebolt & Robert (1994) or Richardson & Green (1997). Stephens (2000*b*) demonstrated that this method failed to separate the two components clearly. The main limitation is that it is very difficult to know which group of parameters should be added to an adequate constraint.

An alternative approach for dealing with the label switching problem is a decision theoretic relabelling strategy. Several studies have used a relabelling approach by finding permutations of the parameters that minimise a loss function (see Stephens (2000*b*), Nobile & Fearnside (2007), Celeux et al. (2000) and Rodríguez & Walker (2014)). This approach, however, have been criticised as computationally expensive for large data sets and for mixture distribution with several components. Another limitation of this method is that it focuses on mixture models with all components having the same type of distributions. for example, mixture of several Normal distributions. For a review on the solutions to label switching problem see Jasra et al. (2005).

In this study therefore, we proposed mixture model with one parametric and one non-parametric components, which does not satisfy the condition for symmetric mixture models. Hence it is asymmetric mixture model. Theoretically, there is no label switching problem for such kind of asymmetric mixture model, since the likelihood is non-exchangeable. The empirical evidence that the method does not incur the label switching problem is fully illustrated in Chapter 4 of this thesis.

2.7 Hidden Markov models

Hidden Markov model (HMM), as an extension of mixture model, is a stochastic process generated by a probabilistic Markov chain with finite number of states, and a set of probability distributions, each associated with its respective state (Koski 2001). Hidden Markov models are popularly applied in many areas as a convenient representation of weakly dependent heterogeneous phenomena (Robert et al. 2000). HMM as a statistical model, originally used for speech recognition (Rabiner 1989), has also been continuously used in numerous applications, such as, modelling economic and financial data, biological sequence analysis and in other areas of artificial intelligence and pattern recognition (Ghahramani 2001).

Hidden Markov model has been adopted in applications, because it suitably provides a formulation for an extension of a mixture model, to allow for spatial data. HMM treats the unobserved latent variable $\boldsymbol{z}$ as a sequence, which has a behaviour of a Markov chain. Here, the latent variable generates the observations $\boldsymbol{x}$ at a time point, and also models transitions between different states of behaviour. The Markov chain property results in the behaviour modelled using a $M \times N$ probability matrix δ, called transition probability matrix, such that the state distribution at each step i, given as z_i is conditioned only on the realisation at previous step $i-1$, given as z_{i-1}. The row of the transition matrix is a probability distribution, which corresponds to the distribution of z_i conditional to z_{i-1},

$$\delta_{ms} = h(z_i = s | z_{i-1} = m), \tag{2.26}$$

and

$$\sum_{s} \delta_{ms} = 1$$

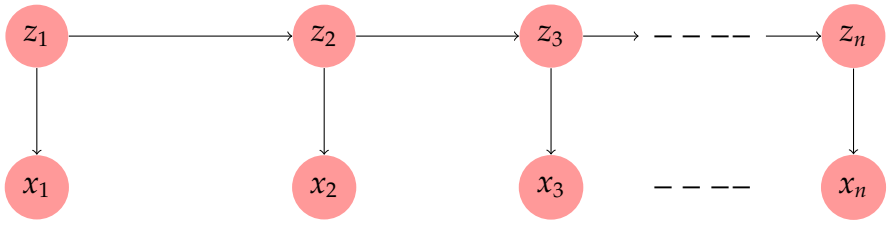


Figure 2.2: Conditional independence relations for Bayesian HMM network (Ghahramani 2001)

A HMM can be explained using a probabilistic directed graph (Figure 2.2) where the nodes show the states and edges representing transitions from one state to another. HMMs require that the observation x_i be drawn independently from a distribution conditional on the latent state z_i. This implies that the dependent behaviour of $\boldsymbol{x}$ is completely attributed to the latent variables $\boldsymbol{z}$. Taking together the distribution of the latent variable and the observations conditioned on the model parameters result in complete data likelihood;

$$l(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{z}) = h(z_1)h(x_1|z_1, \boldsymbol{\theta}) \prod_{i=2}^{n} h(z_i|z_{i-1})h(x_i|z_i, \boldsymbol{\theta}). \tag{2.27}$$

According to Ghahramani (2001) HMM is defined by three properties. Firstly, the observation x_i is generated by some process whose state (unobserved latent variable) z_i is hidden to the observer. Secondly, the state of the hidden process satisfies the Markov property. Thirdly, the latent state is discrete (Ghahramani 2001). Detailed discussions and key references on HMMs are provided by Rabiner (1989) and Scott (2002).

2.7.1 Markov random field model

Markov random field (MRF) is a stochastic process that came originally from statistical physics; a generalisation of Markov processes in which a time index is replaced by a space index (Kindermann & Snell 1980). MRF, as a natural extension to the concept of Markov chain, is a set of random variables described using an undirected graph (Chaudhary 2014), suitable for spatial data. Consider an undirected graphical model in Figure 2.3. Let G be a graph with a set of vertices, each vertex represents a latent state z_i. Let E be a set of edges, each of which connects a pair of vertices. The edges captures the inter dependency among vertices since they are undirected. A clique C is contained in a graph G, where C is a subset of the vertices in G, such that there exists an edge between all pairs of nodes in the subset. As an illustration, given $G = (V, E)$, and z_i, $(i \in V)$ is a MRF define on G (Jung 2009).

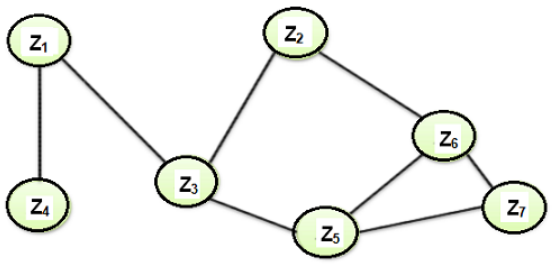


Figure 2.3: An undirected graph to illustrate MRF model (Jung 2009)

From Figure 2.3 we notice that z_3 is conditionally independent of z_4, z_6 and z_7, given z_1, z_2 and z_5. Areas where MRF have been applied includes, computer vision (Lempitsky et al. 2010), machine learning (Salakhutdinov 2009), biological network and genomic data analysis (Bao et al. 2014, Wang et al. 2013, Wei & Li 2007).

2.7.2 Model specification and inference

A Markov chain defined in (2.26) is a first order Markov chain, which satisfies the Markov condition,

$$h(z_i = m|z_{-i}) = h(z_i = m|z_{i-1}, z_{i+1}), \tag{2.28}$$

where $z_{-i} = \{z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n\}$. This is a MRF model which satisfies the first order Markov property. The above definition is symmetric, which means that the full conditional distribution of z_i depends only on the neighbours z_{i-1} and z_{i+1} (Blake et al. 2011).

The model inference and posterior distribution for MRF follows the same case with the finite mixture model for component parameters. But the estimation of $\boldsymbol{z}$ differs greatly with that obtained in the finite mixture model. Because in the MRF, $\boldsymbol{z}$ constitutes a Markov chain with transition probability in (2.28). The full conditionals for $\boldsymbol{z}$ is given as

$$\boldsymbol{z} \sim f(\boldsymbol{z}|\boldsymbol{x}, \theta). \tag{2.29}$$

For more discussions on MRF model, see Chandgotia et al. (2014), Zhang et al. (2001) Bremaud (1999), and Guttorp & Minin (1995). The one-dimensional MRF is implemented in Chapter 6, along with discussions on the posterior distributions.

CHAPTER 3

INTRODUCTION TO CHIP-SEQUENCE DATA

3.1 Introduction

DNA (deoxyribonucleic acid) is the molecule that carries genetic information in almost all organisms (*U.S. National Library of Medicine* 2015). It belongs to a class of molecules called nucleic acids, which are long chains of nucleotides. DNA is made up of two strands of nucleotides, which consists of a sugar phosphate, to which a base is attached. There are four individual building blocks nucleotides in DNA. These are; adenine (A), thymine (T), guanine (G), and cytosine (C). Each nucleotide contains a base. Base pair up naturally only between A and T, and between G and C to form a base pairs unit. Figure 3.1 illustrates the two chains of a double helix.

A gene is a stretch of DNA that contains an instructions to create a molecule called protein in an organism. A complete set of DNA sequence, including all of its genes are the components of genome. A genome in human is arranged into 23 chromosomes. A chromosome consist of long chain of DNA and associated proteins.

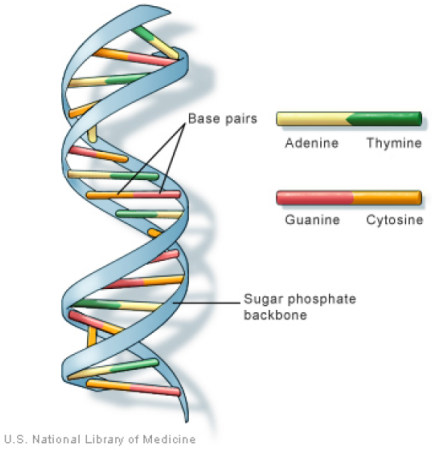


Figure 3.1: Double-stranded DNA (helix)(*U.S. National Library of Medicine* 2015).

DNA and RNA (Ribonucleic acid) are two type of molecules that are responsible for producing genetic information. DNA is the genetic material and RNA is its transcript. Gene expression mechanisms tell us how information contained in a gene translates into a useful product. A gene can be expressed as a protein. There are two major steps for making a protein from a gene within each cell; transcription and translation. In the transcription step, the cell nucleus contains an information stored in a gene's DNA. This information from the gene's DNA is transferred to RNA. The type of RNA that contains the protein-making information is known as messenger RNA (mRNA) (protein-coding genes). The mRNA carries this information out of the nucleus from the DNA into the cytoplasm (*U.S. National Library of Medicine* 2015).

The second step for making a protein from gene take place inside the cytoplasm, and the process is known as translation. The mRNA is "read" based on a specialized complex known as a ribosome, which associates the DNA sequence to the amino acid sequence in proteins. Each sequence of three bases in mRNA forms a codon, and each codon codes for a particular amino acid. Then the transfer RNA (tRNA - a type of RNA), gathers the

protein, one amino acid at a time (*Nature Education* 2014). Figure 3.2 show that through the processes of transcription and translation, information from genes is used to make proteins. Together, the steps of transcription and translation are known as gene expression. *U.S. National Library of Medicine* (2015), Suganuma & Workman (2011), and Nicholl (2008) provide more on biological gene expression processes.

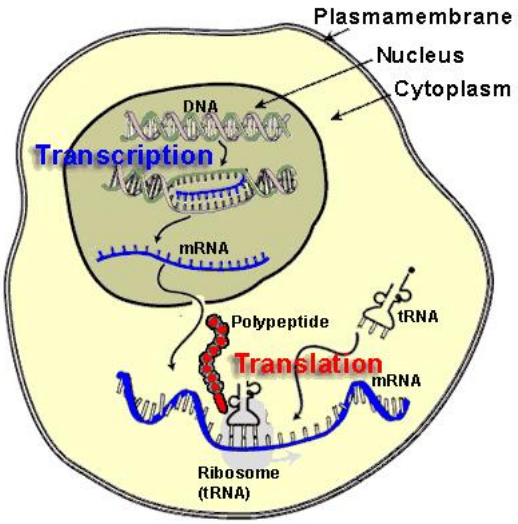


Figure 3.2: Transcription and translation processes for making proteins from genes

Genes are surrounded by DNA sequences that govern their expression. Transcription factors are proteins that regulate gene expression through recognizing and binding to specific DNA sequences (Mo 2012). Another binding protein called histone, is a little cluster of eight proteins (H2A,H2B,H3,H4), two subunits of each protein. They are positively charged, wrapped around DNA through interactions with the negative charges of DNA (Suganuma & Workman 2011). They provide a structural support and contribute immensely in controlling the activities of the genes. Histone proteins can be modified after they are made. For instance, histones can be methylated, acetylated, sumolated, and so on.

In order to fully understand the regulatory network of histone modifications and transcription factors, it is of great significance to study the machinery of these proteins at the

genomic level. Thus, a powerful biological experiment with chromatin immunoprecipitation followed by high-throughput sequencing (ChIP-seq) allow one to locate genome-wide binding sites of a transcription factor, histone modification or DNA methylation (Park 2009). We briefly describe below, the experimental work flow of ChIP-seq.

3.2 Overview of ChIP-seq experiment

Chromatin immunoprecipitation sequencing, also called ChIP-sequencing (ChIP-seq), is a widespread experimental approach in functional genomic and medical research (Schweikert et al. 2013). It is crucial in appreciating many biological processes. The ChIP-seq experiment to identify DNA-protein binding sites is fully discussed in the literature of Lefrançois et al. (2010) and Park (2009). In the ChIP technology, the protein-chromatin are crosslinked to DNA with a cross linking agent, typically formaldehyde. The chromatin is isolated and sheared into smaller fragments. DNA fragments are then co-immunoprecipitated using an antibody that binds specifically to the protein of interest. The crosslinks are then reversed to remove the remaining unbound DNA. Finally, the released DNA is sequenced to identify the genome-wide sites associated with the protein of interest. Figure 3.3 illustrates the procedure. The DNA tags are aligned to the genome. Typically, only the tags that are mapped to the genome are considered for the analyses. Due to the size of the genome, it is natural to divide the entire genome into consecutive base pairs regions. Majority of the regions contain no or fewer tags (noise) and the rest consist of large tags (signal).

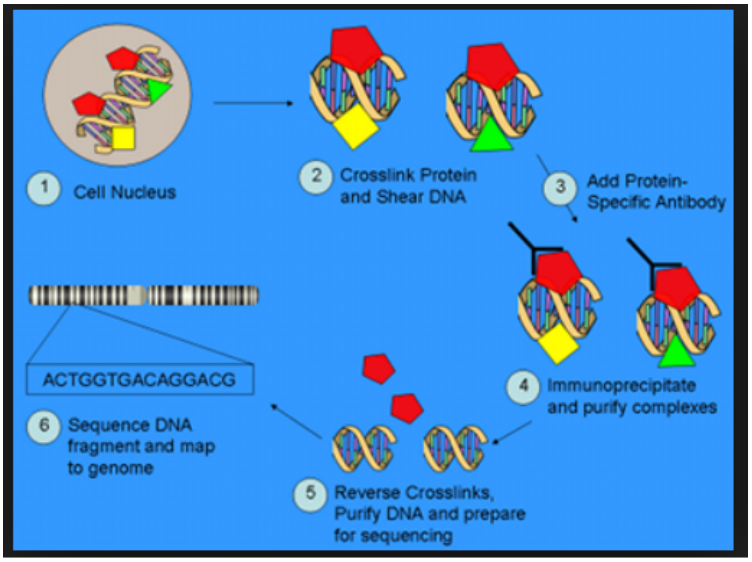


Figure 3.3: ChIP-Seq is used to analyze protein-DNA interactions (Lim 2010)

3.3 ChIP-seq data

In this study we use two proteins, p300 and CBP (CREB-binding protein) generated in Ramos et al. (2010). They belong to a class of binding protein called Histone AcetylTransferras (HATs). HATs is a histone modifying enzyme, which add acetyl groups to target histones, and HATs is associated with transcriptional activation (Wang et al. 2009). The p300 and CBP are transcriptional co-activators largely known to regulate same genes, and are crucial for number of biological functions (see Ramos et al. (2010) and Chen et al. (2013) for biological significance of p300 and CBP). Several ChIP-seq study have been carried out to detect genomic binding sites of p300 and CBP proteins. For example, Ramos et al. (2010) and Wang et al. (2009) found the number of regions bound by p300 and CBP using different antibody specificity for p300 and CBP proteins. Bao et al. (2013) and Bao et al. (2014) employed statistical analysis to detect bound regions for p300 and CBP proteins.

For the purpose of modelling, there is the need to mention some typical features of the

ChIP-seq data sets. Both p300 and CBP are of very high dimension, given the size of the genome. The data sets are discrete in nature and usually consist of an excess of zeroes. The counts of neighbouring windows are typically dependent, especially when the data set consist of smaller window size regions. Further descriptions of the data sets are contained in Vinciotti & Bao (2013), and in the Chapters in which they are analysed.

3.4 Statistical analysis

The final data generated by the experiment report the number of aligned DNA fragments in the sample for each position along the genome. The statistical analysis is aimed at distinguishing the truly enriched regions from the background noise along the genome. Since the genomic regions are either bound by the protein or not bound by the protein, it is therefore a mixture model problem with noise and signal components. Several researchers employed a mixture model approach to analyse ChIP-seq data, with different distributions chosen for modelling the noise component. For example, Mo (2012) and Bao et al. (2013) employed a Poisson distribution for the noise component, Spyrou et al. (2009) and Kuan et al. (2011) used negative Binomial distribution for the noise component and Qin et al. (2010) and Bao et al. (2014) adopted zero-inflated models for the noise component. In this thesis Poisson distribution, negative Binomial diatribution and zero-inflated diatributions (e.g. zero-inflated Poisson or zero-inflated negative Binomial distributions) are used to model the noise component. In the following sections, brief outlines of these parametric distributions, with respect to ChIP-seq data analysis are presented.

3.4.1 Poisson distribution

The Poisson distribution is a discrete probability distribution used to model the number of counts of event occurring randomly within a given time interval. The Poisson probability mass function is given by

$$h(x;\lambda) = \frac{\lambda^x e^{-\lambda}}{x!} \quad x = 0, 1, 2, \ldots, \tag{3.1}$$

where λ is the shape parameter and indicates the average number of occurrences in the given time period. It is the expected number of rare counts in the windows of the genome.

The Poisson distribution is the popular choice for modelling the noise component in ChIP-seq studies, since a region not bound by the protein is a rare event. This was considered, for example by Mo (2012) and Bao et al. (2013), in which they used a Poisson distribution to model the background noise. The choice is appropriate when the regions in the genome consist of 1000 base pairs long. The noise distribution in ChIP-seq data however, may be over-dispersed in relation to the Poisson distribution. The Poisson distributional assumption of equality of mean and variance might not fully capture the complexity in the noise distribution. This is true when smaller window sizes are considered for the regions. In such case, a negative Binomial distribution becomes a better choice for the noise component.

3.4.2 Negative Binomial distribution

The background noise in ChIP-seq data is known to be non-uniform. Some authors abandon the Poisson model in favour of more sophisticated approaches that account for over

dispersion. One popular strategy is to model the noise distribution in windows of certain size as following a negative Binomial distribution. The probability mass function for NB distribution is given as

$$h(x|r,v) = \frac{\Gamma(r+x)}{\Gamma(r)\Gamma(x+1)} v^r (1-v)^x, \tag{3.2}$$

where $\Gamma(\cdot)$ is the Gamma function defined as

$$\Gamma(r) = \int_0^\infty x^{r-1} e^{-x} dx. \tag{3.3}$$

The non-negative dispersion parameter r is the number of successes, and v is the probability parameter for the NB distribution. The NB distribution has the mean

$$E(x) = \mu = r\frac{(1-v)}{v}, \tag{3.4}$$

where

$$r = \frac{\mu^2}{\sigma^2 - \mu}$$

and

$$v = \frac{r}{r+\mu}.$$

The variance is given as

$$\text{var}(x) = \sigma^2 = r\frac{(1-v)}{v^2}. \tag{3.5}$$

We further have that

$$\sigma^2 = \mu + \frac{1}{r}\mu^2 \tag{3.6}$$

Negative Binomial distribution is an over-dispersed Poisson distribution which allows for greater variance. For a large r, NB distribution approaches a Poisson distribution. That is the variance approaches the mean as $r \to \infty$ (Cook 2009).

Majority of the regions in the genome are not enriched, with significantly more empty regions, which give rise to an excess of zeroes in the observed data. This forms the noise distribution and motivates researchers in ChIP-seq data analysis to consider a zero-inflated distribution to model the noise distribution.

3.4.3 Zero-inflated distributions

A zero-inflated model provides a way of modelling the excess of zeroes in addition to allowing for over-dispersion (Lambert 1992). It is a mixture model of two components: a zero mass component (i.e., zero with probability 1) and a count component. Hence, the observed zeroes in the data can come from both sources: either they are "excess" or "structural" zeroes from the first component, or "random" or "sampling" zeroes from the second (count) component. The special mixture model for zero-inflated distribution is given by:

$$h_1(x) = \pi\omega_1(x, 0) + (1 - \pi)\omega_2(x, \boldsymbol{\theta}), \tag{3.7}$$

where $h_1(x)$ is the noise component, π denotes the probability of the structural zeroes, $\omega_1(x, 0)$ is the degenerate distribution at 0 and $\omega_2(x, \boldsymbol{\theta})$ is the count distribution. In ChIP-seq data analysis, two popular zero-inflated models often used to fit the noise distribution are the zero-inflated Poisson (ZIP) and the zero-inflated negative Binomial (ZINB) models.

Zero-inflated Poisson distribution

The probability mass function for ZIP distribution with parameters π and λ is given as,

$$ZIP(x|\pi,\lambda) = \begin{cases} \pi + (1-\pi)e^{-\lambda}, & if\ x = 0 \\ (1-\pi)\frac{e^{-\lambda}}{x!}\lambda^x, & if\ x > 0 \end{cases} \tag{3.8}$$

where $0 \leq \pi \leq 1$ and $\lambda \geq 0$. The parameter π gives the extra probability thrust at the value 0. Note that when $\pi = 0$, then ZIP(π, λ) reduces to Poi(λ) (Beckett et al. 2014). The mean of the ZIP distribution is given by

$$E(x) = (1-\pi)\lambda, \tag{3.9}$$

and has the variance

$$\sigma^2 = (1-\pi)(\lambda + \pi\lambda^2). \tag{3.10}$$

In practice however, the non-zero component in the zero-inflated model may be over-dispersed in relation to the Poisson model. In such a circumstance, the zero-inflated negative Binomial (ZINB) model better accounts for the over-dispersion compared to the ZIP model (Flynn & Francis 2009).

Zero-inflated negative Binomial distribution

The probability mass function for ZINB distribution is given as,

$$ZINB(x|\pi,r,v) = \begin{cases} \pi + (1-\pi)v^r, & if\ x = 0 \\ (1-\pi)\frac{\Gamma(r+x)}{\Gamma(r)\Gamma(x+1)}v^r(1-v)^x, & if\ x > 0, \end{cases} \tag{3.11}$$

where π is defined in (3.8), the dispersion parameter r and the probability parameter v are defined in (3.2). The mean for ZINB is given as,

$$E(x) = (1 - \pi)\mu, \tag{3.12}$$

and the variance is

$$\sigma^2(x) = (1 - \pi)\mu(1 + \mu(\pi + r)), \tag{3.13}$$

where μ is given in (3.4) as,

$$\mu = r\frac{(1 - v)}{v}. \tag{3.14}$$

3.5 Conclusion

Two-component mixture model with Poisson distribution as a choice for the noise component fits the data in ChIP-seq studies very well when 1000 contiguous base pairs long genomic regions are considered. This is implemented in Chapter 5. In cases where smaller window sizes (say, 200bp) are considered for the regions, the counts in the neighbouring windows are spatially dependent. Consequently, the true enriched regions could easily cross some adjacent windows in the genome. Elaborate models which account for Markov properties, such as HMMs or MRF model are required in order to cater for the spatial dependencies, with distributions that caters for over-dispersion, such as NB distribution or zero-inflated distributions as the choice for the noise component. This is implemented in Chapter 6.

CHAPTER 4

MIXTURES OF DISCRETE DISTRIBUTIONS WITH A NON-PARAMETRIC COMPONENT

4.1 Introduction

The density of a typical mixture distribution as written in (2.10) is

$$h(\boldsymbol{x}|\boldsymbol{\pi}, \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k h_k(x_i|\boldsymbol{\theta}_k), \quad \sum_k \pi_k = 1. \tag{4.1}$$

By relaxing distributional assumptions, a mixture model provides a convenient semi-parametric framework for modelling distributions of unknown shape. For example, it is used for model-based density estimation, since any distribution can be approximated by a mixture of elementary components.

In the last two decades, many new methodologies have been proposed for the Bayesian analysis of finite mixture models, such as Diebolt & Robert (1994), West (1997), Richardson & Green (1997), Stephens (2000*a*), McLachlan & Peel (2004) and Nobile & Fearnside (2007).

The existing literature, as described in Chapter 2, have shown that finite mixture models can be inferred in a simple and effective way in a Bayesian estimation framework. Persistent challenges, however, still exist in the diagnostic of Markov Chain Monte Carlo (MCMC) convergence due to the following aspects.

The first aspect is the label switching problem, which is caused by the symmetry in the likelihood function. Many methods exist on how to tackle the label switching problem, for example, there are methods that impose identifiability constraints (Diebolt & Robert 1994, McLachlan & Peel 2004, Richardson & Green 1997) and others that are based on relabelling algorithms (Celeux 1998, Celeux et al. 2000, Rodríguez & Walker 2014, Stephens 2000*b*). For a review and comparison of these methods see, for example, Jasra et al. (2005) and Sperrin et al. (2010). One problem common to the existing methods for dealing with the label switching problem is that they usually require heavy computational costs, which make them unsuitable for large data sets and models with a large number of components. Another drawback of these methods is that they focus on mixture models where all components have the same type of distributions and focus on dealing with the invariance of the likelihood with respect to the permutation of the component labels. When the mixture components have different types of distributions, such as a mixture of Poisson and negative Binomial distributions, label switching problems will still occur, since the likelihood function may still have symmetries. This is because the sampler cannot identify the constraints in the parameter space of the mixture components. The existing methods, however, for dealing with this problem may not be suitable any more.

The second aspect is the identification of the number of components, K. Many authors have devised different methodologies for estimating the number of components in a

Bayesian finite mixture models, for example reversible jump MCMC (Richardson & Green 1997) and Birth and Death MCMC (Nobile & Fearnside 2007, Stephens 2000*a*). Another approach to deal with the unknown number of components is to use a mixture of Dirichlet processes (Antoniak 1974, Escobar & West 1995), which allows for an infinite number of components.

The challenges mentioned above limit the applicability of mixture models in the areas involving large data sets and a large number of components. This motivates our study, as we discuss in detail in the following subsection.

4.1.1 Motivation of the study

In practice, we are often only interested in classifying the observations into two classes. For example, in the analysis of ChIP-seq data, we are interested in whether a region of the genome is bound by the protein in question or not (Bao et al. 2014). There are only two possible classes for such ChIP-seq (discrete) data, but it is inappropriate to use a mixture of two known parametric distributions (e.g. Poisson or negative Binomial distributions). This is because such data sets usually have long tails and the tails may show multi-modal patterns.

For illustration, consider the ChIP-seq data generated by Ramos et al. (2010) for identifying the genomic regions bound by the histone acetyltransferases p300. The data report the number of bound fragments that align to each consecutive region in the genome. A higher value means that the corresponding region is most likely to be bound by the protein in question. Table 4.1 provides the summary statistics for the data set, where we consider the data for 1000 contiguous base pairs long regions along the genome on chromosome

Table 4.1: Summary statistics of the ChIP-seq data of Ramos et al. (2010) for one experiment on the protein p300 on chromosome21.

Sample size	min	max	Mean	Variance
33916	0	282	2.24	18.70

21 (Bao et al. 2013). Figure 4.1 shows a histogram of the count data. The left plot shows that the data set has a very long tail. If we zoom in the tail of the distribution (right plot), we see possible multi-modal patterns, suggesting that the distribution of the data is likely to consist of several component distributions. The interest, however, is that of classifying each region into two possible classes: bound or not bound by the protein in question.

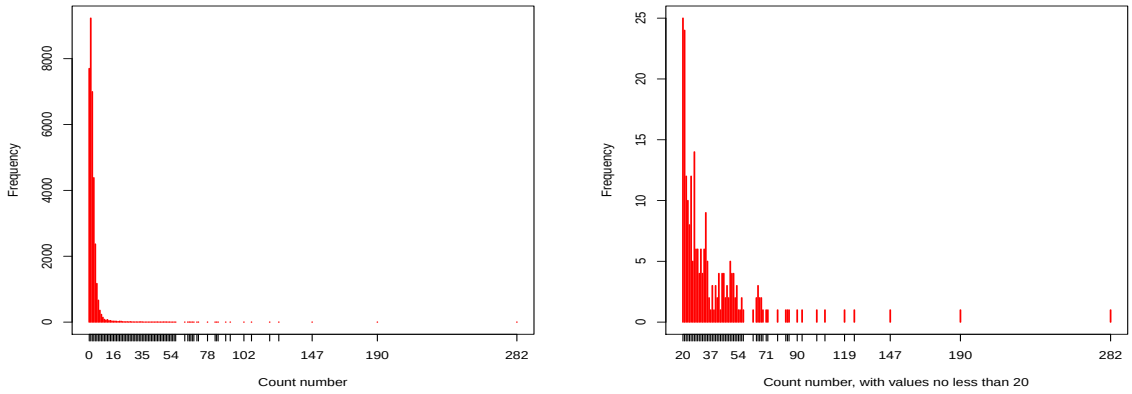


Figure 4.1: Distribution of ChIP-seq data (p300) for one experiment (left), with zoom on the tail (right).

The above situation has been observed also for other ChIP-seq experiments (see Spyrou et al. (2009) and Kuan et al. (2011)), where a two-component parametric mixture model appears to be too restrictive for the analysis of these data. An alternative approach is to use K components, with $K > 2$. In the context of ChIP-seq data analysis, this is considered by Kuan et al. (2011), who allow the signal distribution to be a mixture of two negative Binomial distributions (i.e. $K = 3$). However, it is very challenging to justify what the true value of K is. Although the reversible-jump Markov chain Monte Carlo method (Green

1995) is readily available, the justification of reversible-jump MCMC convergence is non-trivial and it requires heavy computational costs. Another challenge of using K components is that it is non-trivial to determine what the component distributions are. For instance, all components may be chosen as Poisson distributions, or only some components are chosen as Poisson distributions and the others are chosen as negative Binomial (NB) distributions. Finally, since we are only interested in predicting two classes, using a mixture distribution with K components seems unnecessary. The above arguments and the motivating example have led us to consider a two-component mixture model for discrete observations, with one parametric distribution and one non-parametric distribution.

There are some existing non-Bayesian methods based on EM-type algorithms which can deal with a two-component mixture model with one parametric component and one non-parametric component. Those methods, however, require strong assumptions and cannot be applied to this study. For example, Song et al. (2010) proposed a mixture model for sequential clustering of observations. Their approach requires a component with known location parameter and the classification algorithm relies on this center parameter. In our study, the location parameter for the noise component is unknown. Xiang et al. (2014) extended the method of Song et al. (2010) by developing an approach which does not require the location parameter to be known. The asymptotic results were not available for the full model as the identifiability problem was not justified in Xiang et al. (2014). We therefore, focus on the Bayesian approach in this study, where large sample properties for the estimates are not our concern since simulation from the posterior distribution is generally the main task in Bayesian analysis. The challenges of Bayesian analysis for mixture models are the label switching problem and the determination of the number of

components K, and the new method circumvents these challenges.

4.2 The model and the posterior distributions

Suppose that discrete observations $x_1, \cdots, x_n$ are sampled from a mixture of distributions with two components, where one component is the noise distribution and the other component is a signal distribution. We simply use the following density to model the data,

$$h(x) = \pi_1 h_1(x; \boldsymbol{\theta}_1) + \pi_2 h_2(x; \boldsymbol{\theta}_2) \tag{4.2}$$

where h_1 is the parametric distribution for the noise, h_2 is the signal distribution and π_1 and π_2 are the corresponding mixture proportions, respectively.

Given in Section 2.5.2 that z_i $(i = 1, .., n)$ is an indicator or latent variable associated with each observation x_i, i.e. $z_i = k$ $(k = 1, 2)$ means that the observation x_i is from component k. The complete likelihood function for $(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$ given the full data is

$$l(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 | \boldsymbol{x}, \boldsymbol{z}) \propto \prod_{i=1}^{n} \left\{ [\pi_1 h_1(x_i; \boldsymbol{\theta}_1)]^{I[z_i=1]} [\pi_2 h_2(x_i; \boldsymbol{\theta}_2)]^{I[z_i=2]} \right\}. \tag{4.3}$$

The noise distribution h_1 is usually simpler to determine. A Poisson distribution is a natural choice for the noise distribution in ChIP-seq studies, since a genomic region not bound by the protein in question is a rare event. Zero-inflated Poisson distributions have been found to fit the noise distribution very well in cases where small window sizes are considered for the regions, as they account for large number of zeros (Bao et al. 2014). This is considered further in Chapter 6.

The signal distribution can show complicated patterns. As explained in Section 4.1.1, it may be difficult to find a suitable parametric distribution to model h_2. If h_2 is further modelled by a mixture of distributions, it may not be easy to deal with the challenges involved in Bayesian mixture models, such as the label switching problem, determining the number of mixture components and to determine the component distributions. Since we are only interested in distinguishing the signal and the noise, it is not necessary to identify how many components the signal distribution is formed of and what these component distributions are. We therefore, consider to use a non-parametric model for the signal component.

The data is discrete and so we can denote with $x_{(1)}, \cdots, x_{(L)}$ the L distinct values of the observations $x_1, \cdots, x_n$. Define

$$h_2^*(x_{(j)}) = p_j, \ \sum_{j=1}^{L} p_j = 1 \tag{4.4}$$

where p_js ($j = 1, \cdots, L$) are the unknown parameters. Here p_j can be interpreted as the probability of $x = x_{(j)}$ given that x is drawn from the signal component. This can be viewed as a non-parametric distribution. The distribution of x under this model is given by

$$h(x) = \pi_1 h_1(x; \boldsymbol{\theta}_1) + \pi_2 \sum_{j=1}^{L} h_2^*(x) I[x = x_{(j)}]. \tag{4.5}$$

We have the following likelihood function based on the distribution in (4.4), given (x_i, z_i)

$(i = 1, \cdots, n)$,

$$\begin{aligned} l(\boldsymbol{\theta}_1, \boldsymbol{p}, \boldsymbol{\pi}|\boldsymbol{x}, \boldsymbol{z}) &\propto \prod_{i=1}^{n} \left\{ [\pi_1 h_1(x_i; \boldsymbol{\theta}_1)]^{I[z_i=1]} \left[\pi_2 \sum_{j=1}^{L} p_j I[x_i = x_{(j)}] \right]^{I[z_i=2]} \right\} \\ &= \pi_1^{n_1} \pi_2^{n_2} \prod_{i=1}^{n} [h_1(x_i; \boldsymbol{\theta}_1)]^{I[z_i=1]} \cdot \prod_{j=1}^{L} p_j^{\sum_{i=1}^{n} I[z_i=2, x_i=x_{(j)}]} \end{aligned} \tag{4.6}$$

where $n_k = \sum_i I[z_i = k]$, $k = 1, 2$.

If we choose uniform priors for $\boldsymbol{\pi}$ and $\boldsymbol{p}$ and denote the prior for $\boldsymbol{\theta}_1$ as $g(\boldsymbol{\theta}_1)$, we have that $\boldsymbol{\pi}$, $\boldsymbol{p}$ and $\boldsymbol{\theta}_1$ are independent under the posterior distributions. The posterior distribution of $\boldsymbol{\pi}$, in particular, is given by the Beta distribution

$$f(\boldsymbol{\pi}|\boldsymbol{x}, \boldsymbol{z}) \propto \pi_1^{n_1} \pi_2^{n_2} := \text{Beta}\,(\boldsymbol{\pi}; n_1 + 1, n_2 + 1). \tag{4.7}$$

The posterior of $\boldsymbol{p}$ is given by the Dirichlet distribution

$$\begin{aligned} f(\boldsymbol{p}|\boldsymbol{x}, \boldsymbol{z}) &\propto \prod_{j=1}^{L} p_j^{\sum_{i=1}^{n} I[z_i=2, x_i=x_{(j)}]} \\ &:= \text{Dirichlet}\,(\boldsymbol{p}; 1 + \sum_{i=1}^{n} I[z_i = 2, x_i = x_{(j)}]), \end{aligned} \tag{4.8}$$

and the posterior for $\boldsymbol{\theta}_1$ by

$$f(\boldsymbol{\theta}_1|\boldsymbol{x}, \boldsymbol{z}) \propto \prod_{i=1}^{n} [h_1(x_i; \boldsymbol{\theta}_1)]^{I[z_i=1]} g(\boldsymbol{\theta}_1). \tag{4.9}$$

We also have that the posterior probability of z_i given $\boldsymbol{x}, \boldsymbol{\pi}, \boldsymbol{p}$ and $\boldsymbol{\theta}_1$ as

$$\begin{aligned} \Pr(z_i = 1|\boldsymbol{x}, \boldsymbol{\pi}, \boldsymbol{p}, \boldsymbol{\theta}_1) &\propto \pi_1 h_1(x_i; \boldsymbol{\theta}_1) \\ \Pr(z_i = 2|\boldsymbol{x}, \boldsymbol{\pi}, \boldsymbol{p}, \boldsymbol{\theta}_1) &\propto \pi_2 \sum_{j=1}^{L} p_j I[x_i = x_{(j)}]. \end{aligned} \tag{4.10}$$

4.2.1 The interpretation of the model

The signal component in (4.5) can be viewed as a non-parametric component since we allocate a probability to each $x_{(j)}$. The probabilities p_j can be viewed as the empirical probabilities estimated via a sampling approach. It is easy to interpret the idea of this non-parametric component in the following way. If the Poisson component has $\lambda = 5$, say, then the probability that an observation with value 30 comes from the Poisson (noise) component will be very small (about e^{-13}). If the empirical distribution (the signal distribution) tells that $P(X = 30) \approx 0.00001$, then we should indeed classify the observation 30 into the signal component, provided that the component proportion values (π_1 and π_2) are in a similar scale.

The posterior predictive distribution of the new model also has a reasonable interpretation, which is actually linked with the Dirichlet process distribution. If we assume that the latent variable $\boldsymbol{z}$ is known, then the posterior predictive distribution is given by

$$\begin{aligned} & f_{pre}(y|\boldsymbol{x}, \boldsymbol{z}) \\ = & \int_{\boldsymbol{\theta}_1, \boldsymbol{p}, \boldsymbol{\pi}} \left(\pi_1 h_1(y; \boldsymbol{\theta}_1) + \pi_2 \sum_{j=1}^{L} p_j I[y = x_{(j)}] \right) \frac{l(\boldsymbol{\theta}_1, \boldsymbol{p}, \boldsymbol{\pi}|\boldsymbol{x}, \boldsymbol{z}) g(\boldsymbol{\theta}_1)}{c} d\boldsymbol{\theta}_1 d\boldsymbol{p} d\boldsymbol{\pi} \end{aligned}$$

where c, depending on $\boldsymbol{x}, \boldsymbol{z}$, is the normalising constant for the full posterior distribution.

Then we have that

$$
\begin{aligned}
& f_{pre}(y|\boldsymbol{x},\boldsymbol{z}) \propto \\
& \int_{\boldsymbol{\theta}_1,\boldsymbol{p},\boldsymbol{\pi}} \left(\pi_1 h_1(y;\boldsymbol{\theta}_1) + \pi_2 \sum_{j=1}^{L} p_j I[y = x_{(j)}] \right) \pi_1^{n_1} \pi_2^{n_2} \prod_{i=1}^{n} [h_1(x_i;\boldsymbol{\theta}_1)]^{I[z_i=1]} \cdot \prod_{j=1}^{L} p_j^{\sum_{i=1}^{n} I[z_i=2, x_i=x_{(j)}]} g(\boldsymbol{\theta}_1) d\boldsymbol{\theta}_1 d\boldsymbol{p} d\boldsymbol{\pi} \\
\propto \quad & \int_{\boldsymbol{\pi}} \left(\pi_1 [\pi_1^{n_1} \pi_2^{n_2}] \right) d\boldsymbol{\pi} \cdot \int_{\boldsymbol{\theta}_1} h_1(y;\boldsymbol{\theta}_1) \left(\prod_{i=1}^{n} [h_1(x_i;\boldsymbol{\theta}_1)]^{I[z_i=1]} \right) g(\boldsymbol{\theta}_1) d\boldsymbol{\theta}_1 \\
& + \int_{\boldsymbol{\pi}} \left(\pi_2 [\pi_1^{n_1} \pi_2^{n_2}] \right) d\boldsymbol{\pi} \cdot \int_{\boldsymbol{p}} \left(\sum_{j=1}^{L} p_j I[y = x_{(j)}] \right) \prod_{j=1}^{L} p_j^{\sum_{i=1}^{n} I[z_i=2, x_i=x_{(j)}]} d\boldsymbol{p}
\end{aligned}
$$

The posterior predictive distribution can further be written as

$$
\begin{aligned}
& f_{pre}(y|\boldsymbol{x},\boldsymbol{z}) = \\
\propto \quad & \mathbf{E}(\pi_1) \int_{\boldsymbol{\theta}_1} h_1(y;\boldsymbol{\theta}_1) \left(\prod_{i=1}^{n} [h_1(x_i;\boldsymbol{\theta}_1)]^{I[z_i=1]} \right) g(\boldsymbol{\theta}_1) d\boldsymbol{\theta}_1 \\
& + \mathbf{E}(\pi_2) \int_{\boldsymbol{p}} \left(\sum_{j=1}^{L} p_j I[y = x_{(j)}] \right) \prod_{j=1}^{L} p_j^{\sum_{i=1}^{n} I[z_i=2, x_i=x_{(j)}]} d\boldsymbol{p} \\
:= \quad & \mathbf{E}(\pi_1) \cdot \mathbf{E}_1(h_1(y;\boldsymbol{\theta}_1)) + \mathbf{E}(\pi_2) \cdot \sum_{j=1}^{L} I[y = x_{(j)}] \mathbf{E}_2(p_j)
\end{aligned}
$$

where $\mathbf{E}(\pi_1)$ and $\mathbf{E}(\pi_2)$ are the posterior expectation of $\boldsymbol{\pi}$, $\mathbf{E}_1(h_1(y;\boldsymbol{\theta}_1))$ is a posterior expectation conditional on all observations allocated to the first component ($z_i = 1$) and $\mathbf{E}_2(p_j)$ is the posterior expectation for $\boldsymbol{p}$ conditional on all observations allocated to the second component ($z_i = 2$).

Based on (4.8) we know that

$$
\mathbf{E}_2(p_j) = \frac{1 + \sum_i I[z_i = 2, x_i = x_{(j)}]}{L + \sum_i I[z_i = 2]}
$$

and based on (4.7) we know that

$$\mathbf{E}(\pi_k) = \frac{n_k + 1}{n + 2}, \quad k = 1, 2.$$

Then with further calculations we have

$$f_{pre}(y|\boldsymbol{x}, \boldsymbol{z}) \propto \frac{n_1 + 1}{n + 2} \cdot \mathbf{E}_1(h_1(y; \boldsymbol{\theta}_1)) + \frac{n_2 + 1}{n + 2} \cdot \frac{1 + \sum_i I[z_i = 2, x_i = y]}{L + n_2} \tag{4.11}$$

which has a very close connection with the posterior predictive distribution for Dirichlet process distributions in Ferguson (1973).

Suppose that a random sample $X_1, \cdots, X_n$ is from a probability space $(\mathbf{R}, \mathcal{B})$ with a random probability measure $\mathbf{P}$, which is a Dirichlet process with a base measure parameter α. Ferguson (1973) showed that the conditional distribution of $\mathbf{P}$ given $X_1, \cdots, X_n$ is still a Dirichlet process with parameter $\alpha + \sum_i \delta_{X_i}$, where δ_u denotes the measure giving mass one to the point u. Ferguson (1973) used the result to derived the posterior predictive distribution for a new variable Y from $\mathbf{P}$, as

$$\mathbf{P}_{pre}(\cdot|X_1, \cdots, X_n) = \frac{\alpha(\mathbf{R})}{\alpha(\mathbf{R}) + n} \cdot \frac{\alpha(\cdot)}{\alpha(\mathbf{R})} + \left(1 - \frac{\alpha(\mathbf{R})}{\alpha(\mathbf{R}) + n}\right) \frac{\alpha(\cdot) + \sum_i \delta_{X_i}(\cdot)}{\alpha(\mathbf{R}) + n} \tag{4.12}$$

which is a mixture of the prior belief α and the empirical distribution. Comparing (4.11) and (4.12) we can see that the posterior predictive distribution of our model is a mixture of the parametric predictive distribution $\mathbf{E}_1(h_1(y; \boldsymbol{\theta}_1))$ conditional on all observations allocated in the first component, and the empirical distribution conditional on all observations allocated in the second component.

Therefore, under our modelling framework and given all observations x_i with its classification indicator z_i, a new observation can be viewed as from a random probability measure $\mathbf{P}$, which is a Dirichlet process with a base measure parameter proportional to $\mathbf{E}_1(h_1(y;\boldsymbol{\theta}_1))$. Ferguson (1973) uses the base measure parameter α as the prior information and the predictive distribution converges to the empirical distribution as the sample size $n \to \infty$. In our study, the base measure parameter can be viewed as the information for the noise component. The latent variable z_i determines the proportion of samples in each component and it can be sampled via the proposed Gibbs sampler algorithm. If we fix all $z_i = 2$, our model degenerates to Ferguson's Bayesian non-parametric model, which cannot deal with classification since the target distribution is estimated via a non-parametric distribution.

4.2.2 The Gibbs sampler

We can use the Gibbs sampler to draw realisations from the posterior distributions (4.7), (4.8), (4.9) and (4.10) and carry out a Bayesian Monte Carlo analysis. We need to update the unknown parameters and the latent variable $\boldsymbol{z}$ by sampling from the conditional posterior distributions in (4.7), (4.8), (4.9) and (4.10) in order to implement the Gibbs sampler. This leads to Algorithm 4:

Algorithm 4: The proposed method.

Initialization, select, $\boldsymbol{z}^{(0)}, \boldsymbol{\pi}^{(0)}, \boldsymbol{p}^{(0)}$ and $\boldsymbol{\theta}_1^{(0)}$;
Set $m = 1$;
repeat
  for $i = 1$ *to* n **do**
    Update z_i with probability (4.10)
  end
  Update $\boldsymbol{\pi}$ from the posterior in (4.7);
  Update $\boldsymbol{p}$ from the posterior in (4.8);
  Update $\boldsymbol{\theta}_1$ from the posterior in (4.9) ;
  $m = m + 1$
until *enough MCMC steps have been simulated*;

4.3 Simulation studies

4.3.1 Scenario 1

We simulate a data set of $n = 500$ observations from a mixture of Poisson and negative Binomial distributions. The true model is

$$h(x) = \pi_1 \text{Poi}(x; \lambda) + \pi_2 \text{NB}(x; r, v), \tag{4.13}$$

where λ is a Poisson distribution parameter defined in (3.1), r and v are parameters for the negative Binomial distribution defined in (3.2). We choose different values of the true parameters in order to study the performance of our proposed method under different situations. We consider three cases, (a) the means of the two components are far apart, (b) the means of the two components are very close and (c) the means of the two components are neither too close nor too far apart. We choose $\pi_1 = 0.8$, i.e. having a larger proportion for the noise component, to reflect our real ChIP-seq data. We also consider the case where the signal and noise have the same component weights, $\pi_1 = \pi_2 = 0.5$.

The simulation studies are based on 20,000 iterations with 10,000 burn-in iterations,

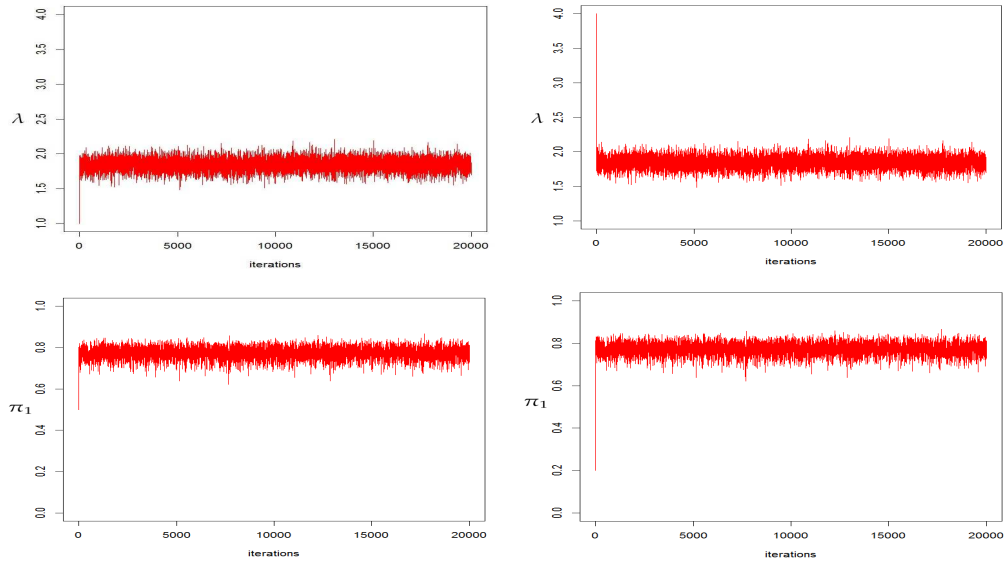


Figure 4.2: Trace plots for π_1 and for λ, with different starting values. The true parameter values are $\pi_1 = 0.8$, $\lambda = 2$, $r = 15$ and $\nu = 0.4$.

repeated for 100 times. Different starting values for the Gibbs sampler are chosen to justify the convergence of the Markov chains. From Figure 4.2 we can see that 20,000 steps are enough to guarantee the convergence for the Markov chains. We also choose different prior distributions to study the sensitivity of our model to the prior used. The results provided in Appendix A demonstrate that the method is robust to different priors.

Table 4.2 show the posterior means of the parameters of the proposed model under a number of different cases. We can see that the estimates are very good when the two components are clearly separated (Set 1 case). There is some bias in the estimate of π_1 for Set 2 and Set 3 as the two component means are very close and many observations from the signal are treated as a sample from the noise component, leading to an inflated estimate of π_1. This kind of bias occurs in all analyses based on mixture models when the component densities are very close, i.e. the two components are not easily identifiable. We can see that label switching does not occur from Figure 4.2. In fact we did not find any label switching

Table 4.2: Simulation results (posterior means and 95% credible intervals) where the true model is (4.13).

	True value			True $\pi_1 = 0.8$		True $\pi_1 = 0.5$	
	λ	r	v	$E(\lambda)$	$E(\pi_1)$	$E(\lambda)$	$E(\pi_1)$
Set 1[a]	6	10	0.3	5.9271	0.7547	6.3299	0.4207
				(5.6054,6.2460)	(0.6895,0.8092)	(5.7362,6.9172)	(0.3362,0.4922)
	10	20	0.2	9.6081	0.7489	9.8108	0.4061
				(8.9084,10.1492)	(0.6723,0.8114)	(9.0310,10.3848)	(0.3322,0.4665)
	2	15	0.4	1.8425	0.7722	1.9963	0.4171
				(1.6765,2.0060)	(0.7136,0.8202)	(1.7091,2.2801)	(0.3355,0.4828)
Set 2[b]	2	5	0.6	2.0375	0.9300	2.3493	0.8285
				(1.8412,2.2088)	(0.8344,0.9823)	(2.0519,2.6615)	(0.7175,0.9115)
	4	2	0.4	3.6699	0.8019	3.2550	0.7131
				(3.1780,4.0766)	(0.6320,0.9201)	(2.5142,3.9011)	(0.5582,0.8262)
	6	5	0.5	5.6248	0.8854	5.3592	0.7631
				(5.2943,5.9334)	(0.7899,0.9552)	(4.6386,5.9127)	(0.6155,0.8681)
Set 3[c]	1	7	0.6	1.0527	0.8276	1.2148	0.5316
				(0.8823,1.2257)	(0.7379,0.8961)	(0.8289,1.9641)	(0.3769,0.6317)
	2.5	6	0.5	2.7537	0.8969	3.0378	0.7282
				(2.5479,2.9584)	(0.8171,0.9479)	(2.6840,3.4131)	(0.6246,0.8061)
	3	5	0.4	3.2014	0.8828	3.7587	0.7137
				(2.9778,3.4199)	(0.8151,0.9313)	(3.3226,4.2288)	(0.6093,0.7937)

[a]Component means are far apart
[b]Component means are close
[c]Components means are neither too close nor too far apart

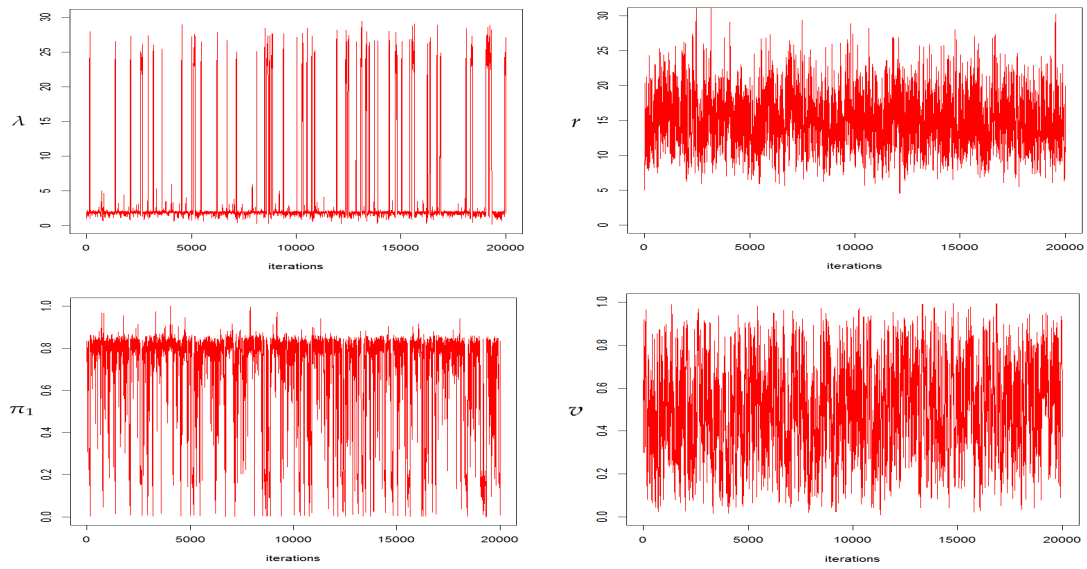


Figure 4.3: MCMC trace plots for λ, π_1, r and v by using a mixture of Poisson and NB distributions; the true model is (4.13) with $\lambda = 2$, $\pi_1 = 0.8$, $r = 15$ and $v = 0.4$.

in the trace plots based on all simulations in Table 4.2. The label switching problem still exists if we use a mixture model with a Poisson component and a NB component (the true underlying model) to analyse the data, although the two components are different. This is shown in Figure 4.3, which is the simulation results for a mixture with two components: one Poisson component with a small mean value 2 and an NB component with a larger mean value around 22.5. We can see from the trace plots that in this case the MCMC chain manages to estimate λ and π_1 close to their true values, 2 and 0.8 respectively. The algorithm, however, sometimes returns estimates of λ around 20 and very small estimates of π_1, meaning that the Poisson distribution is used to model observations with large values but the NB distribution is used to model observations with small values. Such a label switching is due to the fact that the algorithm can not identify the constraints in the parameter space of the mixture component, making the likelihood function symmetry, and makes it impossible to draw conclusions from the MCMC chains without some form of

relabelling. All existing methods, however, cannot deal with such label switching problems since they require the component distributions to be of the same type. Here we cannot simply relabel a Poisson parameter say $\lambda = 20$ to the pair of NB parameters (r, v). For simplicity of presentation we did not provide any results (such as posterior means and credible intervals) based on a mixture of Poisson and NB distribution here, since those results are severely biased.

4.3.2 Scenario 2

We now consider a more general mixture distribution with five-components, where the noise component is a Poisson distribution and the signal components are NB distributions. The sample size is also chosen as $n = 500$. The aim here is to show that our method outperforms the fully parametric mixture model, under general mixture distributions, in terms of estimation and classification. The true model for this simulation is given by

$$h(x) = \pi_1 \text{Poi}(x; \lambda) + \sum_{k=2}^{5} \pi_k \text{NB}(x; r_k, v_k). \tag{4.14}$$

We chose different values for the parameters λ, r_k and v_k in order to compare our method with existing methods under different settings. In the first case, we choose the set of true parameters (Set 1) as $\lambda = 2$, $\pi_1 = 0.6$, $\pi_2 = \cdots = \pi_5 = 0.1$, $\boldsymbol{r} = (15, 13, 10, 8)$ and $\boldsymbol{v} = (0.9, 0.7, 0.6, 0.5)$. This choice of $\boldsymbol{r}$ and $\boldsymbol{v}$ for the NB components gives the corresponding component means as $(1.68, 5.57, 6.67, 8.00)$. Such a choice implies that the means of Poisson component and all the other NB components are not too far apart. From Table 4.3 we can see that our method has clear posterior estimates, which approximate the true parameter

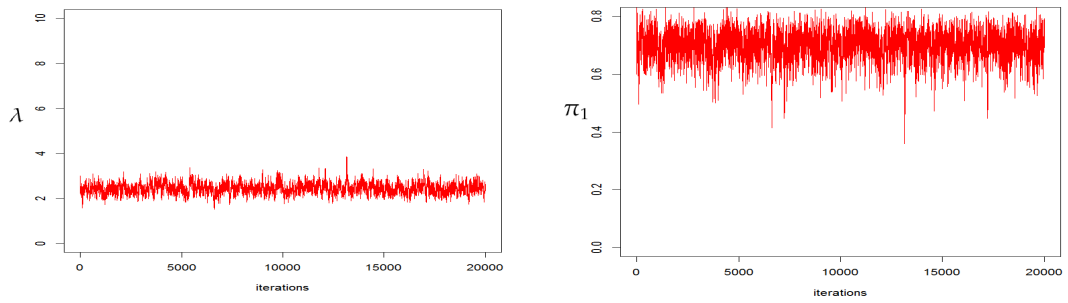


Figure 4.4: MCMC trace plots for λ, π_1 for our new model for the true parameter values in Table 4.3

value. The trace plot confirms that our method does not suffer from the label switching problem (see Figure 4.4).

However, for the Poisson component and other NB components, the above situation causes some identifiability problems when traditional Gibbs sampling method is used. The MCMC trace plots in Figure 4.5 for π_1 and λ clearly show the occurrence of the label switching problem. This issue severely distorts the posterior estimates, see Table 4.3. For example the posterior mean for λ is 2.4371 (the true value is 2) and the posterior mean for π_1 is 0.2952 (the true value is 0.6). On the contrary, if we use the proposed method, the estimates for λ and π_1 are 2.2514 and 0.6987, respectively, which are closer to the true values. We did not provide the estimates for $\boldsymbol{r}$ and $\boldsymbol{v}$ since the main aim here is classification and under the new model $\boldsymbol{r}$ and $\boldsymbol{v}$ are not involved. Instead we compare the misclassification rate (the ratio of the number of wrongly classified observations over the total number of observations) for the two methods. This can be easily obtained as the Bayesian approach provides the simulated $\boldsymbol{z}$ from the full posterior. From the last column of Table 4.3 we can see that our method has smaller misclassification rate than the parametric mixture model.

In the second set of the simulation the choice of the true parameters are $\lambda = 7$, $\pi_1 = 0.6$, $\pi_2 = \cdots = \pi_5 = 0.1$, $\boldsymbol{r} = (15, 20, 40, 30)$ and $\boldsymbol{v} = (0.4, 0.3, 0.3, 0.2)$. This choice of $\boldsymbol{r}$ and $\boldsymbol{v}$ for

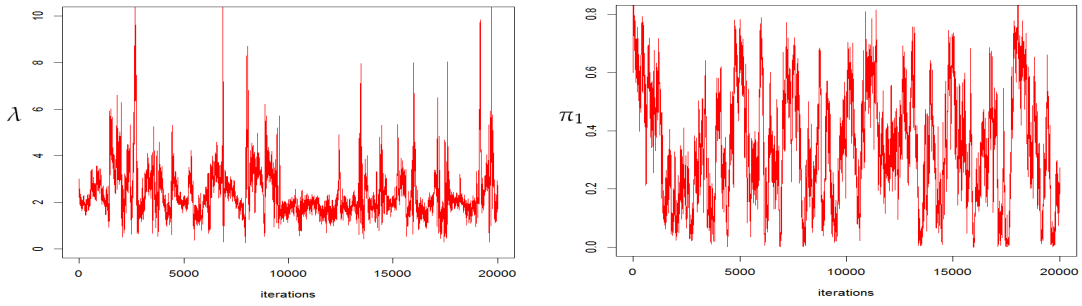


Figure 4.5: MCMC trace plots for λ, π_1 for a mixture of a Poisson and four NB distributions for the true parameter values in Table 4.3

Table 4.3: Parameter Set 1. (i) the new method; (ii) true mixture model of five components.

Model	True value										Posterior mean		Error rate
	λ	π_1	r_1	r_2	r_3	r_4	v_1	v_2	v_3	v_4	$E(\lambda)$	$E(\pi_1)$	e
(i)	2	0.6	15	13	10	8	0.9	0.7	0.6	0.5	2.2514 (1.8881,2.6680)	0.6987 (0.5680,0.7885)	0.31
(ii)	2	0.6	15	13	10	8	0.9	0.7	0.6	0.5	2.4371 (1.0576,4.9958)	0.2952 (0.0249,0.7433)	0.46

the NB components gives the corresponding component means as $(22.5, 46.7, 93, 120)$. This gives very different component means with the Poisson component having the smallest mean. This situation is similar to the real ChIP-seq data in terms of the long tail and the noise component having the smallest mean value. From Table 4.4 we can see that our method gives posterior mean estimate for π_1 with smaller bias and shorter credible interval than the parametric mixture approach, and our method gives competitive result with the fully parametric mixture model when estimating the λ. The larger bias and variation in the estimates in the existing methods contrarily, is due to the label switching problem, see Figure 4.6. The new method still performs better in terms of classification rate (see Table 4.4).

We ran the Gibbs sampler for 20,000 steps with 10,000 steps as burn-in iterations over 100 simulations. We further use a Metropolis-Within-Gibbs sampler to simulate from the posterior distributions for the parametric mixtures given the difficulty in simulating the

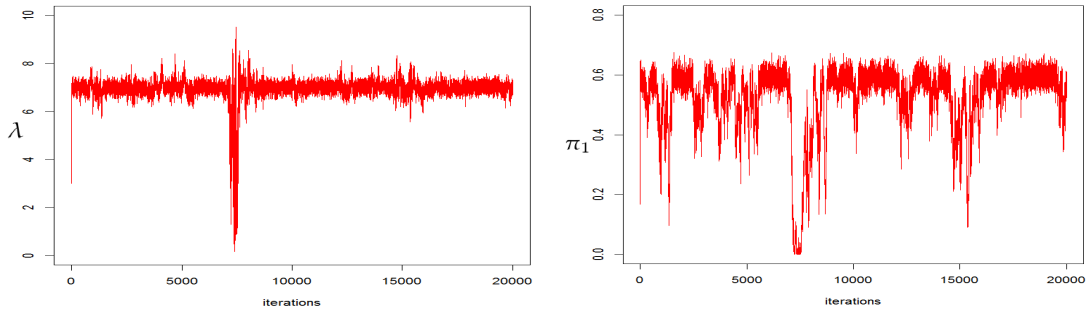


Figure 4.6: MCMC trace plots for λ, π_1 for a mixture of a Poisson and four NB distributions for the true parameter values in Table 4.4

Table 4.4: Parameter Set 2. (i) the new method; (ii) the true mixture model of five components.

Model	True value										Posterior mean		Error rate
	λ	π_1	r_1	r_2	r_3	r_4	v_1	v_2	v_3	v_4	$E(\lambda)$	$E(\pi_1)$	e
(i)	7	0.6	15	20	40	30	0.4	0.3	0.3	0.2	6.8676 (6.4998,7.2305)	0.5787 (0.5226,0.6292)	0.06
(ii)	7	0.6	15	20	40	30	0.4	0.3	0.3	0.2	6.9622 (6.4599,7.4080)	0.5349 (0.2279,0.6329)	0.10

parameters $\boldsymbol{r}$ and $\boldsymbol{v}$ for NB distributions.

4.3.3 Scenario 3

We still consider the true model in (4.14) and generate $n = 500$ observations. The aim here is to further justify the classification performance of our methodology under different settings. Just as in the second scenario, we choose the set of true parameters (Set 1) as $\lambda = 1$, $\pi_1 = 0.6$, $\pi_2 = \cdots = \pi_5 = 0.1$, $\boldsymbol{r} = (3, 5, 8, 10)$ and $\boldsymbol{v} = (0.3, 0.5, 0.7, 0.8)$. This choice of $\boldsymbol{r}$ and $\boldsymbol{v}$ for the NB components gives the corresponding component means as $(7, 5, 3.43, 2.5)$. Such a choice imply that the means of all components are not too far away, with Poisson component having the smallest mean value. We can use the posterior probability distribution for $\boldsymbol{z}$ as the classification criteria to justify the classification performance of the new method. The

posterior probability of $z_i = 1$ is given by

$$g_i = P(z_i = 1|\boldsymbol{x}, \boldsymbol{\theta}) := \frac{\pi_1 h_1(x_i; \boldsymbol{\theta}_1)}{\pi_1 h_1(x_i; \boldsymbol{\theta}_1) + \pi_2 \sum_{j=1}^{L} p_j I[x_i = x_{(j)}]}. \tag{4.15}$$

If g_i is less than a threshold, say ρ, the value x_i will be classified into class 2. Based on this idea, the false discovery rate (FDR) is commonly used to justify the performance of a classifier and was for example used by Bao et al. (2013) in the context of mixture models. It is defined as

$$\begin{aligned} FDR &= \frac{\#\{\text{false positive discovery}\}}{\#\{\text{declared positive}\}} \\ &= \frac{\#\{\text{false positive discovery}\}}{\sum_i I[g_i < \rho]}. \end{aligned} \tag{4.16}$$

We fixed the FDR at level 0.01 and find the threshold ρ and further calculate the false non-discovery rate (FNDR) based on the existing method and our new proposed method. The FNDR is defined as

$$FNDR = \frac{\#\{\text{false negative discovery}\}}{\#\{\text{declared negative}\}}$$

The FNDR values are shown on the last column of Table 4.5. The new method has smaller FNDR.

We ran the Gibbs sampler for 20,000 steps with 10,000 steps as burn-in iterations for all the simulation results. We choose a Gamma$(2, 1)$ prior distribution for λ and a uniform prior distribution for $\boldsymbol{\pi}$ for both methods. We choose uniform priors for $\boldsymbol{p}$ for the new method, whereas we choose a Gamma$(20, 1)$ prior for the elements of $\boldsymbol{r}$ and a uniform

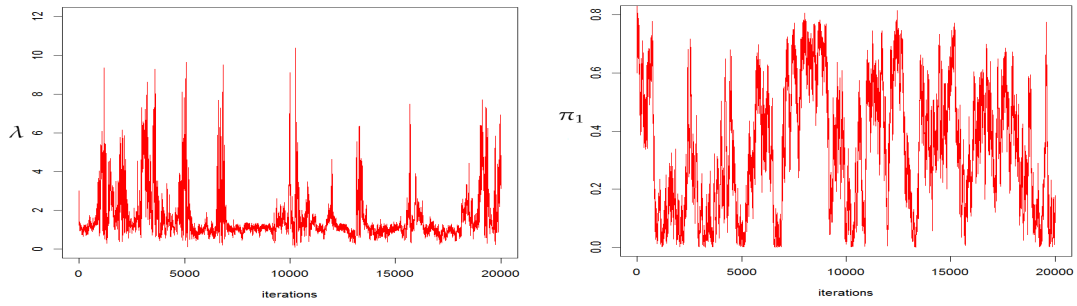


Figure 4.7: MCMC trace plots for λ, π_1 by using the true model, a mixture of a Poisson and four NB distributions for the true parameter values in Table 4.5.

Table 4.5: Parameter Set 1. (i) the new method; (ii) existing mixture model, the true mixture model of five components is used. FDR is controlled at level 0.01.

Model	True value										Posterior mean		FNDR
	λ	π_1	r_1	r_2	r_3	r_4	v_1	v_2	v_3	v_4	$E(\lambda)$	$E(\pi_1)$	
(i)	1	0.6	3	5	8	10	0.3	0.5	0.7	0.8	1.3516	0.7488	0.078
											(1.0855,1.6115)	(0.6385,0.8260)	
(ii)	1	0.6	3	5	8	10	0.3	0.5	0.7	0.8	1.9227	0.3101	0.406
											(0.6301,3.6701)	(0.0368,0.7504)	

distribution for the elements in $\boldsymbol{v}$ for the Poisson-NB mixture. The choice of Gamma prior for $\boldsymbol{r}$, since it can be viewed as a shape parameter of Gamma distribution (see (3.3)). Gamma prior for $\boldsymbol{r}$ has been used, for example, by Bradlow et al. (2002). As in Scenario 2, a Metropolis-Within-Gibbs sampler is used to simulate from the posterior distributions for the parametric mixture models given the difficulty in simulating the parameters $\boldsymbol{r}$ and $\boldsymbol{v}$ for the NB distributions.

In a second simulation (Set 2) we consider the following true parameters as $\lambda = 5$, $\pi_1 = 0.6$, $\pi_2 = \cdots = \pi_5 = 0.1$, $\boldsymbol{r} = (5, 7, 10, 14)$ and $\boldsymbol{v} = (0.4, 0.6, 0.8, 0.9)$. This choice of $\boldsymbol{r}$ and $\boldsymbol{v}$ for the NB components gives the corresponding component means as $(7.5, 4.67, 2.5, 1.56)$. Such a choice still gives very close means for each component but now the Poisson component does not have the smallest mean. The posterior estimates based on the traditional

parametric mixture model are still very poor and our method returns a smaller FNDR (see Table 4.6).

Table 4.6: Parameter Set 2. (i) the new method; (ii) existing mixture model, the true mixture model of five components is used. FDR is controlled at level 0.01.

Model	True value										Posterior mean		FNDR
	λ	π_1	r_1	r_2	r_3	r_4	v_1	v_2	v_3	v_4	$E(\lambda)$	$E(\pi_1)$	
(i)	5	0.6	5	7	10	14	0.4	0.6	0.8	0.9	4.5530	0.8010	0.24
											(4.1507,4.9451)	(0.6720,0.8930)	
(ii)	5	0.6	5	7	10	14	0.4	0.6	0.8	0.9	4.1706	0.4278	0.52
											(0.6884,5.7709)	(0.0082,0.8059)	

In the final simulation we choose the set of true parameters (Set 3) as $\lambda = 6$, $\pi_1 = 0.6$, $\pi_2 = \cdots = \pi_5 = 0.1$, $\boldsymbol{r} = (8, 12, 30, 40)$ and $\boldsymbol{v} = (0.3, 0.3, 0.4, 0.3)$. This choice of $\boldsymbol{r}$ and $\boldsymbol{v}$ for the NB components gives the corresponding component means as $(18.7, 28, 45, 93.3)$. Just as in set 3 of the second scenario such a choice will give very different component means with the Poisson component having the smallest mean. As argued in the second scenario, the choice of the true parameter values gives similar situation to the real ChIP-seq data in terms of tailed distribution. From Table 4.7 we can see that our method gives posterior mean estimate for π_1 with smaller bias and shorter credible interval than the parametric mixture approach. The new method gives competitive result for the estimate of λ (see Table 4.7). Once again, the larger bias and variation in the estimates given by the existing methods is due to the label switching problem. Since the signal and the noise components are far apart, in this case however, the new method did not gain an advantage in terms of FNDR, when controlling the FDR at level 0.01.

Table 4.7: Parameter Set 3. (i) the new method; (ii) existing mixture model, the true mixture model of five components is used. FDR is controlled at level 0.01.

Model	True value										Posterior mean		FNDR
	λ	π_1	r_1	r_2	r_3	r_4	v_1	v_2	v_3	v_4	$E(\lambda)$	$E(\pi_1)$	
(i)	6	0.6	8	12	30	40	0.3	0.3	0.4	0.3	5.7662	0.5799	0.10
											(5.4113,6.1190)	(0.5227,0.6325)	
(ii)	6	0.6	8	12	30	40	0.3	0.3	0.4	0.3	5.8718	0.5372	0.03
											(5.0773,6.5490)	(0.0548,0.6376)	

4.4 Conclusion

We developed a mixture model with a parametric component for modelling noise and a non-parametric component for modelling signal. The new method can still distinguish whether an observation is signal or noise, which is the main research interest in the studies that we consider, and it can do so with higher accuracy than a mixture of two parametric distributions, since it fits the data better. We showed several advantages for using a non-parametric distribution for the signal component. Firstly, we neither need to specify the distributions for the signal component nor to consider how many components there are. Secondly, the method does not incur the label switching problem. Results on simulated data verify the validity of the approach and show a better performance of the method compared to fully parametric mixture distributions under general cases.

In Chapter 5, we show the applicability of the new method to ChIP-seq data on two Histone AcetylTransferases (HATs) proteins, p300 and CREB binding protein (CBP) for a single experiment, with the aim to identify enriched gene regions. The performance of our new method is assessed by comparing with parametric models.

CHAPTER 5

ANALYSIS OF CHIP-SEQ DATA VIA BAYESIAN FINITE MIXTURE MODELS WITH A NON-PARAMETRIC COMPONENT

5.1 Introduction

Chromatin immunoprecipitation followed by next generation sequencing (ChIP-Seq) is an efficient process for genome-wide profiling of DNA-protein interactions. ChIP-seq technologies have became a popular tool in biomedical research for studying transcription factors binding sites and histone modifications (Park 2009). As a result of large amount of sequence tags and the complexities of the signal, statistical analysis of ChIP-seq data poses a great challenge (Mo 2012). Several approaches have been proposed for the analysis of ChIP-seq data with the aim of identifying genome-wide binding sites. Some approaches involved the use of non-parametric methods and focus mainly on peak calling algorithms (see Nix et al. (2008), Zhang et al. (2008) and Wang et al. (2010)). Other attempts also exist for the analysis of ChIP-seq data using latent mixture model approach by assuming

a parametric signal distribution mixed with a parametric noise distribution. Kuan et al. (2011) for example, propose a mixture of negative Binomial distributions for the signal component and a negative Binomial distribution for the noise component, and Bao et al. (2014) propose a zero-inflated Poisson/NB distributions for noise and a NB distribution for the signal. Section 4.1.1 provided the limitations of parametric mixture model approach. In this study, therefore, we consider Bayesian mixture model approach with a parametric and a non-parametric components.

5.2 Motivation

For illustration, we use ChIP-seq data generated by Ramos et al. (2010) for the experiments on p300 and CREB binding protein (CBP) for identifying the genomic regions bound by the histone acetyltransferases. The data report the number of bound fragments that align to each region in the genome. For the CREB binding protein (CBP), the data set consist of 33,916 regions. The lowest count is zero, imply a region is not bound by the protein of interest, and the highest count is 214, means a particular region is bound by enough protein of interest. The mean and the variance are 2.13 and 8.76 respectively. Consider Table 4.1 and Figure 4.1 for the distribution and histogram of the count data for p300 protein. The plot shows that the data set has a very long tail. When the plot is zoomed (right plot), the tailed distribution shows possible multi-modal patterns, suggesting that the distribution of the data is likely to consist of several component distributions. One can adopt the Poisson distribution as a choice for the noise component, since the count is a rare event. But the signal distribution show a complicated pattern. We therefore consider using

a non-parametric distribution for the signal component.

5.3 The method

Suppose that the discrete observations $x_1, \cdots, x_n$ are sampled from a mixture of distributions with two components, where one component is the noise distribution and the other component is a signal distribution. The complete likelihood function for $(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$ conditional on the full data as given in (4.3) is

$$l(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 | \boldsymbol{x}, \boldsymbol{z}) \propto \prod_{i=1}^{n} \left\{ [\pi_1 h_1(x_i; \boldsymbol{\theta}_1)]^{I[z_i=1]} [\pi_2 h_2(x_i; \boldsymbol{\theta}_2)]^{I[z_i=2]} \right\}. \tag{5.1}$$

In order to develop a non-parametric distribution for the signal component h_2, denote $x_{(1)}, \cdots, x_{(L)}$ as the L distinct values of the observations $x_1, \cdots, x_n$, and let p_j be defined in (4.4), where p_js ($j = 1, \cdots, L$) are the unknown parameters. Based on (4.4), the distribution of $\boldsymbol{x}$ under the non-parametric component is given in (4.5) as

$$h(x) = \pi_1 h_1(x; \boldsymbol{\theta}_1) + \pi_2 \sum_{j=1}^{L} h_2^*(x) I[x = x_{(j)}], \tag{5.2}$$

with likelihood function in (4.6), given (x_i, z_i) as

$$\begin{aligned} l(\boldsymbol{\theta}_1, \boldsymbol{p}, \boldsymbol{\pi} | \boldsymbol{x}, \boldsymbol{z}) &\propto \prod_{i=1}^{n} \left\{ [\pi_1 h_1(x_i; \boldsymbol{\theta}_1)]^{I[z_i=1]} \left[\pi_2 \sum_{j=1}^{L} p_j I[x_i = x_{(j)}] \right]^{I[z_i=2]} \right\} \\ &= \pi_1^{n_1} \pi_2^{n_2} \prod_{i=1}^{n} [h_1(x_i; \boldsymbol{\theta}_1)]^{I[z_i=1]} \cdot \prod_{j=1}^{L} p_j^{\sum_{i=1}^{n} I[z_i=2, x_i=x_{(j)}]}; \end{aligned}$$

where $n_k = \sum_i I[z_i = k]$, $k = 1, 2$.

Based on the posterior distributions in (4.7), (4.8), (4.9) and (4.10), the Gibbs sampler for our proposed method in Algorithm 4 can be employed to draw realisations.

5.4 Data analysis

We now show the applicability of the new method to ChIP-seq data. In a ChIP-seq experiment the DNA is sheared into smaller fragments that are then sequenced. The final data generated by the experiment report the number of aligned DNA fragments in the sample for each position along the genome. Due to noise and the size of the genome, it is common to summarise the raw counts by dividing the genome into consecutive regions, typically with a length between 200 and 1000 base pairs. The datasets considered in this analysis are the p300T301.1000bp and the CBPT301.1000bp from the R package `enRich` (Bao & Vinciotti 2013), which are both size-selected into 1000 base pairs. For more description of the ChIP-seq technology and these particular datasets see Chapter 3 and the references therein. The aim of the analysis is to detect the regions in the genome bound by the histone acetyltransferases p300 and CBP, and so it is a two-component mixture model problem with a background noise and a signal components.

The posterior classification probability in (4.15) can be computed based on the posterior distributions to predict whether a region is enriched or not. The region i will be classified as an enriched region if $g_i < \rho$. The threshold value ρ is determined by controlling the false discovery rate at a predefined level (Bao et al. 2014), say 0.001. We controlled the FDR at the smallest value of 0.1% in order to exclude genomic regions that showed unstructured and anomalous read counts from the analysis. The expected false discovery rate corresponding

to the threshold value ρ is given in (4.16) by

$$0.001 = \widehat{FDR} \quad = \quad \frac{\sum_{i \in \text{enriched region}}(g_i)}{\sum_i I[g_i < \rho]}.$$

Figure 5.1 show Venn diagrams of the regions detected as enriched by p300 and CBP using the model proposed, compared with a mixture of two Poisson distributions and a mixture of two NB distributions at 0.1% false discovery rate. For the Poisson and NB mixtures we use the implementation in the `enRich` R package (Bao & Vinciotti 2013). At the same FDR, our method detects more enriched regions than the existing methods. In order to validate the enriched regions identified by the three methods, we use ChromHMM (Ernst & Kellis 2010). In other words, further analysis is done to ascertain that the additional enriched regions identified are not just false positives but likely functional transcriptional activators. These regions are also assessed for the presence of Transcription Start Sites (TSSs) of annotated genes and other chromatin features using ChromHMM (Ernst & Kellis 2010). Figure 5.2 illustrates the results based on ChromHMM with 3 chromatin states. The top plots demonstrates the emission probabilities in respect of the various analyses, which indicates the observed enrichment probability, conditional on each of the three states. The plots therefore, tend to indicate that majority of the identified enrichment pattern were explained by two of the three states. In addition, the relative fold enrichment for various annotations are presented in the bottom plot. In general, the pattern in which these two states are enriched with active and weak promoters, strong enhancers and TSSs are revealed in the plots. The plots further illustrate how the second state, which is mainly identified by our method, reflects a larger degree of enrichment of active and weak promoters. We can,

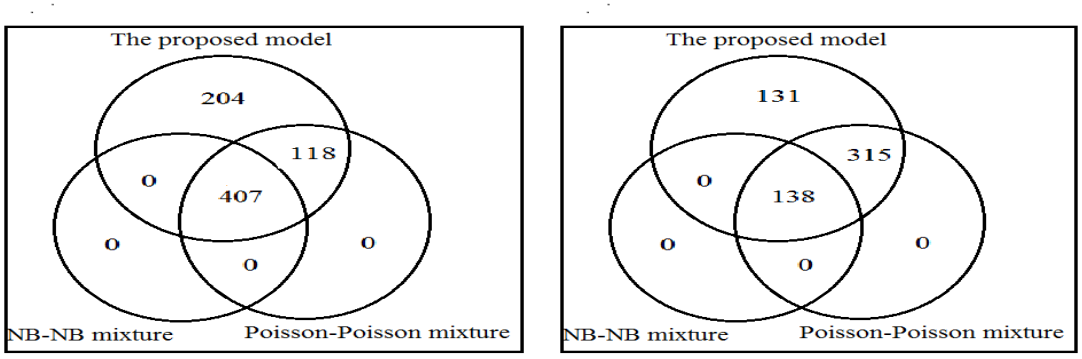


Figure 5.1: Number of enriched regions identified by the proposed model, Poisson-Poisson mixture model and NB-NB mixture model for p300 (left plot) and CBP (right plot) datasets on chromosome21 at the 0.1% FDR.

therefore, conclude that under the same FDR, our proposed method detect more regions which are generally of the same quality with the ones found by existing methods.

5.5 Conclusion

In this Chapter we illustrated the proposed method on ChIP-seq data to detect the enriched regions bound by the proteins in question. The value of L was not too large, in the scale of 100 for the data sets analyzed in this Chapter, and therefore the method was efficient. The method may not be practical, however, if L is up to several thousands. In the context of ChIP-seq data one solution for this is to consider smaller window sizes for genomic regions, e.g. 200 base pairs long, which will automatically reduce the value of L. When smaller genomic regions are considered, however, the true enriched regions could easily cross two or more adjacent windows. In this case, the spatial dependencies between neighbouring windows along the genome should be taken into account. A large proportion of the genome is expected to be not enriched, resulting in an excess of zeros in the noise component for smaller genomic regions. In order to account for spatial dependencies

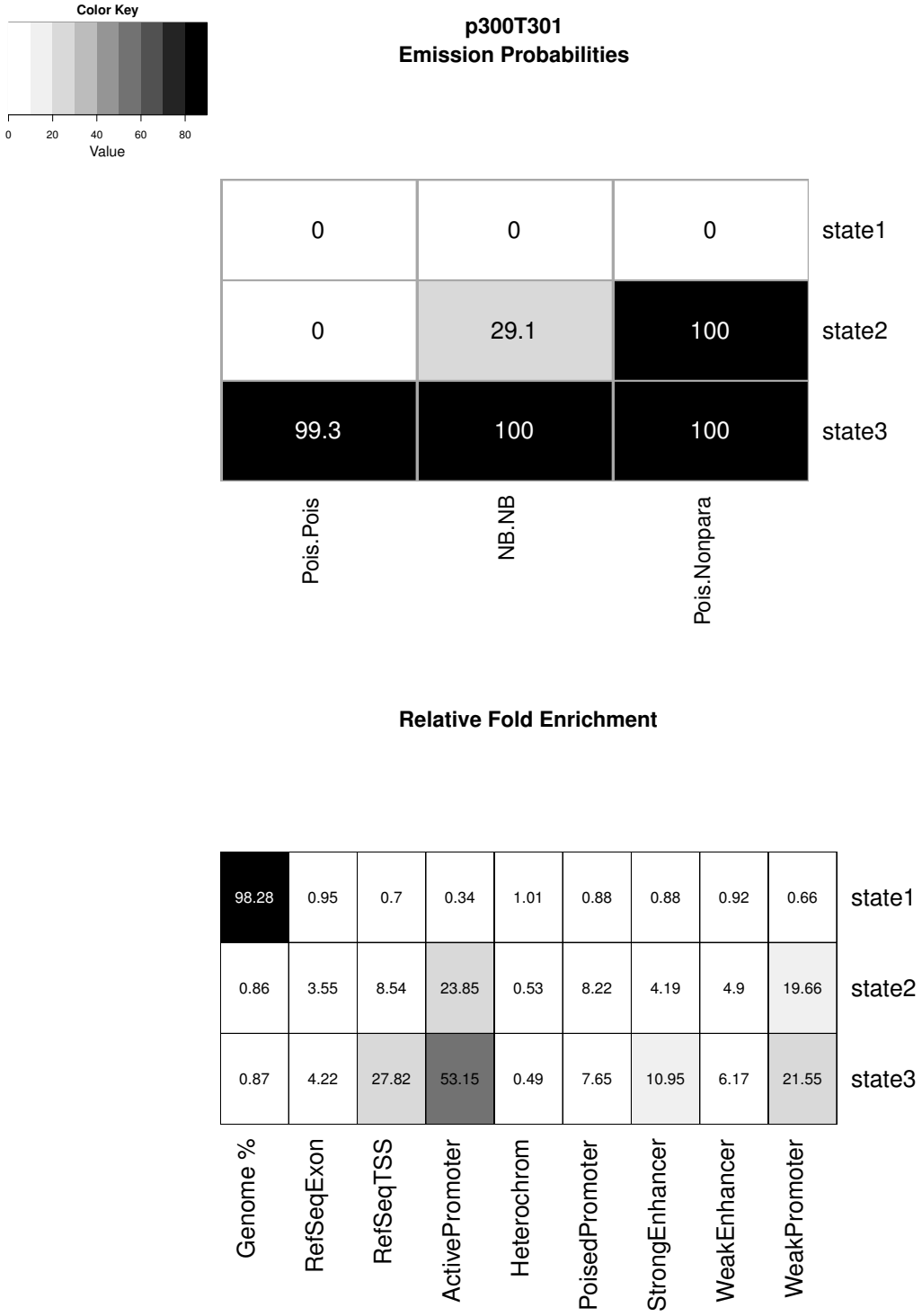


Figure 5.2: Validation of the enriched bins detected. The top plots show heatmaps of the probabilities (in percentages) that the p300 detected bins are enriched given each identified chromatin-state. The bottom plot shows the relative percentage of the genome represented by each chromatin state (first column) and the relative fold enrichment for several types of annotation (remaining columns).

along the genome and the excess of zeros, zero-inflated models (e.g. zero-inflated Poisson or zero-inflated negative Binomial) are a better choice for the noise component combined with more elaborate models which account for Markov properties, such as HMMs or Markov random fields, of the type developed by Spyrou et al. (2009) and Bao et al. (2014). The extension of the proposed methodology is implemented in Chapter 6.

CHAPTER 6

MARKOV RANDOM FIELD MODEL FOR THE ANALYSIS OF MIXTURES OF DISCRETE DISTRIBUTIONS WITH A NON-PARAMETRIC COMPONENT

6.1 Introduction

The output from ChIP-seq experiments report the number of aligned DNA fragments in the sample for each position along the genome. The total count is summarized by dividing the genome arbitrarily into consecutive fixed-size regions of length 200 base pairs long (Bao et al. 2014). For such window sizes, as argued in Chapter 3, the counts of neighbouring windows are typically correlated. Consequently, there is expectation that the data will have spatial dependencies between neighbouring windows along the genome. The true enriched regions may cross over some adjacent windows. There is the need, therefore, to adopt statistical models with a Markov property that entails using the transitions between hidden states to model the spatial relationship. In addition, majority of the regions in the genome are expected not to be enriched with a significantly larger proportion of empty

regions. This form the noise distribution and motivates researchers in ChIP-seq studies to consider more elaborate distributions to model the noise component.

In the context of ChIP-seq studies, several researchers adopted HMM-based methodologies to detect the enriched genomic regions with different distributions chosen for the mixtures. For example, Qin et al. (2010) used a ZIP distribution for the noise component and a generalized Poisson distribution for the signal component. Spyrou et al. (2009) used negative Binomial (NB) distribution for the noise component, and the sum of two NB distributions for the signal component. Bao et al. (2014) proposed ZIP or ZINB distributions for noise component and Poisson or NB distributions for the signal component.

We argued in Chapter 4 that it is too restrictive to use a parametric mixture model of two-component distribution, taking into account the nature of the signal distribution for a ChIP-seq data (see Figure 4.1), which is likely to consist of several component distributions. If a mixture of several components are used for the signal distribution, it is computationally very difficult to deal with the challenges involved in Bayesian analysis for mixture models, like label switching problem and determining the number of components. In this study, therefore, we proposed the use of one-dimensional Markov random field (MRF) model with a NB distribution or a zero-inflated distribution (ZIP or ZINB distributions) for the noise component and the proposed non-parametric distribution for the signal component. The proposed MRF model account for the spatial dependencies in the observations. The methodology is described below.

6.2 Methods

6.2.1 A one-dimensional MRF model

In this section we describe one-dimensional Markov random field model as implemented in Bao et al. (2014). The latent variable z_i satisfies one-dimensional Markov properties given in (2.28) as

$$h(z_i = m|z_{-i}) = h(z_i = m|z_{i-1}, z_{i+1}), \quad m \in \{1, 2\}, \tag{6.1}$$

where $z_{-i} = \{z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n\}$. The Markov assumption in (6.1) leads to a classical two states Markov chain defined as

$$\begin{aligned} h(z_1, \cdots, z_n) &= \delta_0(z_1) \prod_{i=1}^{n-1} \delta z_i, z_{i+1} \\ &= \delta_0(z_1) \delta_{2,2}^{n_{2,2}} \delta_{2,1}^{n_{2,1}} \delta_{1,2}^{n_{1,2}} \delta_{1,1}^{n_{1,1}}, \end{aligned} \tag{6.2}$$

where $\delta_0(z_1)$ is the initial state distribution and $\delta_{m,s} = h(z_{i+1} = s|z_i = m)$; $m, s \in \{1, 2\}$ are the transition probabilities. This is a classical one-dimensional Markov random field model, often referred to as a hidden Markov model (HMM). Unlike the above model, in this study, the joint density of the latent variables can be presented as

$$h(z_1, \ldots, z_n) = \frac{\prod_{i=1}^{n-1} h(z_i, z_{i+1})}{\prod_{i=2}^{n-1} h(z_i)}, \tag{6.3}$$

where $h(z_i, z_{i+1})$ is the joint probability of z_i and z_{i+1}, and $h(z_i)$ is the marginal probability of z_i. And we also have that $h(z_i) = \sum_{z_{i+1}} h(z_i; z_{i+1})$. For binary variables, $z_i = m \in \{1, 2\}$, as in our

case, model (6.3) becomes

$$h(z_1,\ldots,z_n) = {q_1}^{I(z_1=1)} q_2^{I(z_1=2)} \left(\frac{q_{1,1}}{q_1}\right)^{n_{1,1}} \left(\frac{q_{1,2}}{q_1}\right)^{n_{1,2}} \left(\frac{q_{2,1}}{q_2}\right)^{n_{2,1}} \left(\frac{q_{2,2}}{q_2}\right)^{n_{2,2}}, \tag{6.4}$$

where $q_{m,s} = h(z_i = m, z_{i+1} = s), m, s \in \{1,2\}, i = 1,\ldots,n-1,$ $n_{m,s} = \#\{z_i = m, z_{i+1} = s\}$, $q_2 = h(z_i = 2) = q_{2,1} + q_{2,2}$, $q_1 = h(z_i = 1) = 1 - q_2$ and $q_{1,2} = q_{2,1}$. Since $\sum_{m,s\in\{1,2\}} q_{m,s} = 1$, then $q_{1,2} = q_{2,1} = (1 - q_{2,2} - q_{1,1})/2$.

We can show that the model in (6.4) satisfies the Markov property in (6.1) and is a one-dimensional MRF model. We can also show that the initial state distribution under (6.3) is the stationary distribution. The quantities $q_{m,s}$ can further be written in terms of transition probabilities satisfying $\delta_{m,s} = \frac{q_{m,s}}{q_m}$ as

$$h(z_1,\ldots,z_n) = \left(\frac{\delta_{2,1}}{\delta_{1,2}+\delta_{2,1}}\right)^{I(z_1=1)} \left(\frac{\delta_{1,2}}{\delta_{1,2}+\delta_{2,1}}\right)^{I(z_1=2)} \delta_{1,1}^{n_{1,1}} \delta_{1,2}^{n_{1,2}} \delta_{2,1}^{n_{2,1}} \delta_{2,2}^{n_{2,2}}. \tag{6.5}$$

Therefore, (6.5) can be viewed as a one-dimensional MRF model with initial state distribution as the stationary distribution.

6.2.2 Parameter Estimation

6.2.2.1 Negative Binomial distribution for the noise component

Let $\delta_{2,2}$ and $\delta_{1,2}$, as defined in (6.2), be the probability that the current state is 2 (enriched) given that the left of it is 2 (enriched) and 1 (not enriched) respectively. We consider a NB distribution for the noise component and the non-parametric distribution for the signal component, and the parameters to be estimated are $\Theta = (\boldsymbol{\theta}_1, \delta_{2,2}, \delta_{1,2})$. The joint likelihood

function given the latent states $z_1, \ldots, z_n$ and the data $x_1, \ldots, x_n$ is given as

$$
\begin{aligned}
l(\Theta, \boldsymbol{p}|\boldsymbol{z}, \boldsymbol{x}) &= h(\boldsymbol{z}|\Theta)h(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{p}, \Theta) \qquad (6.6)\\
&\propto \left(\frac{\delta_{1,2}}{\delta_{1,2}+1-\delta_{2,2}}\right)^{I(z_1=2)} \left(\frac{1-\delta_{2,2}}{\delta_{1,2}+1-\delta_{2,2}}\right)^{I(z_1=1)} (\delta_{2,2})^{n_{2,2}}(1-\delta_{2,2})^{n_{2,1}}(\delta_{1,2})^{n_{1,2}}(1-\delta_{1,2})^{n_{1,1}}\\
&\times \prod_{i=1}^{n}\left[\frac{\Gamma(r_1+x_i)}{\Gamma(r_1)\Gamma(x_i+1)} v_1^{r_1}(1-v_1)^{x_i}\right]^{I[z_i=1]} \cdot \prod_{j=1}^{L} p_j^{\sum_{i=1}^{n} I[z_i=2, x_i=x_{(j)}]}.
\end{aligned}
$$

6.2.2.2 Zero-inflated distribution for the noise component

Zero-inflated distributions allow one to model the mass zero and count distribution separately. The zero-inflated Poisson distribution or the zero-inflated negative Binomial distribution is a mixture of degenerate zero mass distribution and a Poisson distribution or a NB distribution. Let y_i be an inner latent variable, such that $h(y_i = 1|z_i = 1) = \pi$, where π is defined in (3.7). The likelihood functions for ZIP distribution and ZINB distribution are described below.

Zero-inflated Poisson distribution

We can estimate the parameters $\Theta = (\boldsymbol{\theta}_1, \delta_{2,2}, \delta_{1,2})$ assuming we consider a ZIP distribution for the noise component and the proposed non-parametric distribution for the signal component. The joint likelihood function given the latent states $\boldsymbol{z}$, the inner latent variables $\boldsymbol{y}$

and the observations $\boldsymbol{x}$, therefore, is given by

$$\begin{aligned} l(\Theta, \boldsymbol{p}|\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}) &= h(\boldsymbol{z}|\Theta)h(\boldsymbol{y}|\boldsymbol{z}=1,\Theta)h(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{y}, \boldsymbol{p}, \Theta) \\ &\propto \left(\frac{\delta_{1,2}}{\delta_{1,2}+1-\delta_{2,2}}\right)^{I(z_1=2)} \left(\frac{1-\delta_{2,2}}{\delta_{1,2}+1-\delta_{2,2}}\right)^{I(z_1=1)} (\delta_{2,2})^{n_{2,2}}(1-\delta_{2,2})^{n_{2,1}}(\delta_{1,2})^{n_{1,2}}(1-\delta_{1,2})^{n_{1,1}} \\ &\times \prod_{i=1}^{n}\left[\pi^{I[y_i=1,x_i=0]}\left((1-\pi)\frac{e^{-\lambda}}{x_i!}\lambda^{x_i}\right)^{I[y_i=2]}\right]^{I[z_i=1]} \cdot \prod_{j=1}^{L} p_j^{\sum_{i=1}^{n} I[z_i=2,x_i=x_{(j)}]} \end{aligned} \tag{6.7}$$

Zero-inflated NB distribution

If we assume that the noise component follows a ZINB distribution and the signal component follows the non-parametric distribution, then the following parameters can be estimated: $\Theta = (\boldsymbol{\theta}_1, \delta_{2,2}, \delta_{1,2})$, where $\boldsymbol{\theta}_1$ are the parameters for the ZINB distribution defined in (3.11). The likelihood function given the latent states $\boldsymbol{z}$, the inner latent variables $\boldsymbol{y}$ and the observations $\boldsymbol{x}$ is given by

$$\begin{aligned} l(\Theta, \boldsymbol{p}|\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}) &= h(\boldsymbol{z}|\Theta)h(\boldsymbol{y}|\boldsymbol{z}=1,\Theta)h(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{y}, \boldsymbol{p}, \Theta) \\ &\propto \left(\frac{\delta_{1,2}}{\delta_{1,2}+1-\delta_{2,2}}\right)^{I(z_1=2)} \left(\frac{1-\delta_{2,2}}{\delta_{1,2}+1-\delta_{2,2}}\right)^{I(z_1=1)} (\delta_{2,2})^{n_{2,2}}(1-\delta_{2,2})^{n_{2,1}}(\delta_{1,2})^{n_{1,2}}(1-\delta_{1,2})^{n_{1,1}} \\ &\times \prod_{i=1}^{n}\left[\pi^{I[y_i=1,x_i=0]}\left((1-\pi)\frac{\Gamma(r+x_i)}{\Gamma(r)\Gamma(x_i+1)}v^r(1-v)^{x_i}\right)^{I[y_i=2]}\right]^{I[z_i=1]} \cdot \prod_{j=1}^{L} p_j^{\sum_{i=1}^{n} I[z_i=2,x_i=x_{(j)}]} \end{aligned} \tag{6.8}$$

6.2.2.3 The posterior and Gibbs sampler

The likelihood functions for the proposed method, when the noise component follows different distributions, such as NB, ZIP, and ZINB distributions have now been introduced. This section now discuses the posterior and the Gibbs sampler for the three distributions.

If we assume a uniform prior for $\delta_{2,2}$, $\delta_{1,2}$ and $\boldsymbol{p}$, and assume the prior for $\boldsymbol{\theta_1}$ as $g(\boldsymbol{\theta}_1)$,

the full conditionals for the parameters $\delta_{2,2}$ and $\delta_{1,2}$ are proportional to

$$\begin{aligned} f(\delta_{2,2}|\boldsymbol{x},\boldsymbol{z}) &\propto h(\boldsymbol{x},\boldsymbol{z}|\delta_{2,2})g(\delta_{2,2}) \\ &\propto (\delta_{2,2})^{n_{2,2}}(1-\delta_{2,2})^{n_{2,1}} \cdot (\delta_{2,2})^{\alpha-1}(1-\delta_{2,2})^{(\beta-1)} \\ &\propto (\delta_{2,2})^{\alpha+n_{2,2}-1}(1-\delta_{2,2})^{\beta+n_{2,1}-1} \\ &:= \text{Beta}(\delta_{2,2}; 1+n_{2,2}, 1+n_{2,1}), \end{aligned} \tag{6.9}$$

$$\begin{aligned} f(\delta_{1,2}|\boldsymbol{x},\boldsymbol{z}) &\propto h(\boldsymbol{x},\boldsymbol{z}|\delta_{1,2})g(\delta_{1,2}) \\ &\propto (\delta_{1,2})^{n_{1,2}}(1-\delta_{1,2})^{n_{1,1}} \cdot (\delta_{1,2})^{\alpha-1}(1-\delta_{1,2})^{(\beta-1)} \\ &\propto (\delta_{1,2})^{\alpha+n_{1,2}-1}(1-\delta_{1,2})^{\beta+n_{1,1}-1} \\ &:= \text{Beta}(\delta_{1,2}; 1+n_{1,2}, 1+n_{1,1}), \end{aligned} \tag{6.10}$$

and the posterior for $\boldsymbol{\theta}_1$ given in (4.9) by

$$f(\boldsymbol{\theta}_1|\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) \propto \prod_{i=1}^{n} [h_1(x_i;\boldsymbol{\theta}_1)]^{I[z_i=1]} g(\boldsymbol{\theta}_1). \tag{6.11}$$

The posterior of $\boldsymbol{p}$ is a Dirichlet distribution as given in (4.8)

$$\begin{aligned} f(\boldsymbol{p}|\boldsymbol{x},\boldsymbol{z}) &\propto \prod_{j=1}^{L} p_j^{\sum_{i=1}^{n} I[z_i=2, x_i=x_{(j)}]} \\ &:= \text{Dirichlet}(\boldsymbol{p}; 1+\sum_{i=1}^{n} I[z_i=2, x_i=x_{(j)}]). \end{aligned} \tag{6.12}$$

The full conditional distribution from which we draw the latent states $\boldsymbol{z}$ (Bao et al. 2014,

Scott 2002) is

$$\Pr(z_i = m|z_{-i}, \boldsymbol{x}, \boldsymbol{p}, \Theta) \quad \propto \quad h(x_i|z_i = m, \boldsymbol{p}, \Theta)h(z_{i-1}, m)h(m, z_{i+1}), \tag{6.13}$$

where the normalizing constant is the sum over all possible values of m. The full conditional distribution from which we draw the inner latent variable y_i, given the latent variable $z_i = 1$ is given by

$$\Pr(y_i = m|z_i = 1, \boldsymbol{x}, \Theta, \boldsymbol{p}) \quad \propto \quad h(x_i|z_i = 1, y_i = m, \Theta, \boldsymbol{p})h(y_i = m|z_i = 1). \tag{6.14}$$

The Bayesian Gibbs sampling technique can be used to draw realisations from their conditional distributions to estimate the parameters based on all the posterior distributions. The Metropolis-within-Gibbs technique is carried out to estimate the parameters for NB and ZINB distributions, just like the case in Chapter 4. The sampling scheme is summarise in Algorithm 5.

Algorithm 5: Gibbs sampler for the proposed model.

Initialization, select, $\boldsymbol{z}^{(0)}$, $\boldsymbol{y}^{(0)}$, $\boldsymbol{\theta}_1^{(0)}$, $\boldsymbol{p}^{(0)}$, $\delta_{2,2}^{(0)}$, and $\delta_{1,2}^{(0)}$;
Set $m = 1$;
repeat
  for $i = 1$ *to* n **do**
    Update z_i with probability in (6.13);
    Update y_i with probability in (6.14);
  end
  Update $\delta_{2,2}$ and $\delta_{1,2}$ from the posterior in (6.9) and (6.10) respectively;
  Update $\boldsymbol{\theta}_1$ from the posterior in (6.11);
  Update $\boldsymbol{p}$ from the posterior in (6.12);
  $m = m + 1$
until *enough MCMC steps have been simulated*;

6.3 Simulation studies

This section discusses simulation studies with different distributions chosen for the noise component and the non-parametric distribution for the signal component. The aim is to justify the convergence of the Markov chain. The simulation framework is presented in three scenarios. In each of these three scenarios, two sets are considered: set 1 and set 2, based on, when the means of the two components are not too far apart and when they are far apart.

6.3.1 Scenario 1

A sample of $n = 500$ observations is drawn from a Markov mixture model of two NB distributions, where $\delta_0 = (0.5, 0.5)$, $\delta_{1,2} = 0.2$ and $\delta_{2,2} = 0.7$.

In set 1 of this simulation study, the means of the two components are considered not clearly separated. First, we chose the following true parameters; $(r_1, v_1) = (3, 0.2)$ and $(r_2, v_2) = (5, 0.2)$. These true parameters give the corresponding means for the two components as $(12, 20)$. In the second case, the true parameter values chosen are; $(r_1, v_1) = (5, 0.6)$ and $(r_2, v_2) = (10, 0.5)$, and these give the means for the two components as, $(3.33, 10)$. And finally, we chose the true parameters as $(r_1, v_1) = (5, 0.4)$ and $(r_2, v_2) = (7, 0.3)$. These give the corresponding means for the two components as $(7.5, 16.33)$ respectively. In all the cases, the noise component having a smaller means.

The observations are analysed using MRF model with NB distribution and the non-parametric distribution. The simulation studies for the two sets (sets 1 and 2) are based on 20,000 MCMC steps discarding the first 10,000 steps as burn-in iterations, repeated for 100

times. Different starting values for the Gibbs sampler are used to justifying the convergence of the Markov chains. From Figure 6.1 we can see that the 20,000 MCMC steps are enough to guarantee the convergence for the Markov chains. We present the results of set 1 on Table 6.1. The results show similarity with the simulation on Table 4.2, in terms of bias in the estimate. We posit that the bias in the estimates happen in most mixture related models when the component densities are too close. We calculate the error rate (ratio of incorrectly classified observations to the total number of observations) to justify the performance of the model in terms of classification.

Table 6.1: The posterior means (with 95% credible intervals) and error rate for set 1 where the true model is a Markov mixture model of two NB distributions.

True value				Posterior mean			Error rate
r_1	r_2	v_1	v_2	$E(r_1)$	$E(v_1)$	$E(\delta_{1,2})$	$E(\delta_{2,2})$	e
3	5	0.2	0.2	3.5097	0.2197	0.4695	0.4705	0.44
				(2.6094,4.6835)	(0.1677,0.2831)	(0.3877,0.5544)	(0.3803 0.5581)	
5	10	0.6	0.5	6.2322	0.5784	0.4652	0.5080	0.39
				(3.0733,11.0559)	(0.3987,0.7480)	(0.3780,0.5559)	(0.4150,0.5977)	
5	7	0.4	0.3	5.0968	0.3778	0.4639	0.4909	0.38
				(3.3177,7.6712)	(0.2741,0.4954)	(0.3812,0.5500)	(0.4023,0.5768)	

The set 2 of this study considers when the components means are far apart. We chose the true parameters for the first simulation in set 2 as $(r_1, v_1) = (5, 0.6)$ and $(r_2, v_2) = (12, 0.1)$. These return the corresponding means for the two components as $(3.33, 108)$. Again, the true parameters chosen for the second case are $(r_1, v_1) = (7, 0.8)$ and $(r_2, v_2) = (14, 0.2)$. These give the means for the two components as $(1.75, 56)$. And finally, the true parameter values chosen for the third case are $(r_1, v_1) = (10, 0.7)$ and $(r_2, v_2) = (15, 0.3)$. These give the corresponding means of $(4.29, 35)$. Such choices of true parameters give very different component means, with the noise component having a smaller mean value. Table 6.2 show

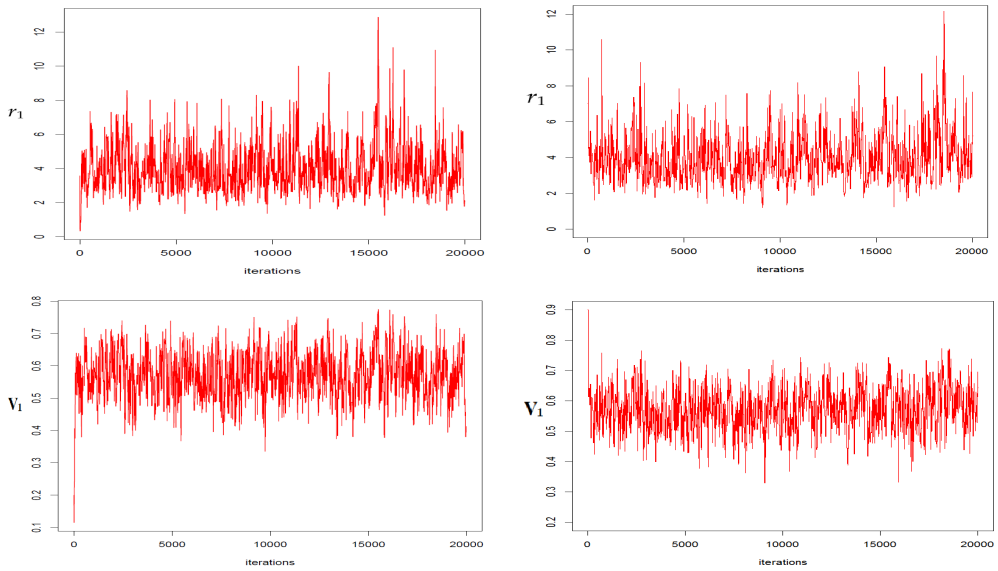


Figure 6.1: Trace plots for r_1 and for v_1, with different starting values. The true parameter values are $r_1 = 0.4$, $v_1 = 0.6$, $r_2 = 25$ and $v_2 = 0.2$.

the simulation results for the different true parameters in the model. It revealed that the estimates are very good when the component means are clearly separated. The algorithm also returns a smaller classification errors.

Table 6.2: The posterior means (with 95% credible intervals) and error rate for set 2 where the true model is a Markov mixture model of two NB distributions.

True value	Posterior mean				Error rate
r_1 r_2 v_1 v_2	$E(r_1)$	$E(v_1)$	$E(\delta_{1,2})$	$E(\delta_{2,2})$	e
5 120.60.1	5.4363	0.6355	0.2941	0.6640	0.09
	(3.5952,7.9964)	(0.5374,0.7288)	(0.2249,0.3702)	(0.5964,0.7292)	
7 140.80.2	7.4998	0.8258	0.3004	0.6695	0.09
	(4.5709,11.4066)	(0.7453,0.8862)	(0.2297,0.3786)	(0.6022,0.7341)	
10150.70.3	10.5023	0.7158	0.3217	0.6631	0.11
	(7.0839,14.8326)	(0.6309,0.7881)	(0.2477,0.4023)	(0.5955,0.7281)	

6.3.2 Scenario 2

In this scenario, 500 observations are generated from a Markov mixture model of ZIP and NB distributions, where $\delta_0 = (0.1, 0.9)$, $\delta_{1,2} = 0.3$ and $\delta_{2,2} = 0.8$.

In Set 1 the means of the two components are not well separated. First, the true parameter values $(\lambda, \pi) = (5, 0.4)$ and $(r_2, v_2) = (15, 0.4)$ provide a corresponding means for the two components as $(3, 22.5)$. Again, we chose the true parameter values $(\lambda, \pi) = (2, 0.3)$ and $(r_2, v_2) = (5, 0.2)$. These give the means as $(1.4, 20)$. And finally, for the true values $(\lambda, \pi) = (1, 0.5)$ and $(r_2, v_2) = (19, 0.5)$ provide a corresponding means of $(0.5, 19)$. The posterior mean estimates for the true parameters are presented on Table 6.3.

Table 6.3: Set 1 simulation results (posterior means and 95% credible intervals) where the true model is a Markov mixture model of ZIP and NB distributions.

True value				Posterior mean				Error rate
λ	π	r_2	v_2	$E(\lambda)$	$E(\pi)$	$E(\delta_{1,2})$	$E(\delta_{2,2})$	e
5	0.4	15	0.4	4.9522	0.3664	0.3449	0.7923	0.04
				(4.4469,5.4917)	(0.2987,0.4368)	(0.2745,0.4196)	(0.7424,0.8382)	
2	0.3	5	0.2	2.2703	0.2857	0.3617	0.7720	0.05
				(1.9126,2.6555)	(0.2093,0.3631)	(0.2922,0.4353)	(0.7198,0.8203)	
1	0.5	19	0.5	0.9113	0.4163	0.3190	0.8227	0.008
				(0.6346,1.2574)	(0.2563,0.5530)	(0.2514,0.3913)	(0.7786,0.8628)	

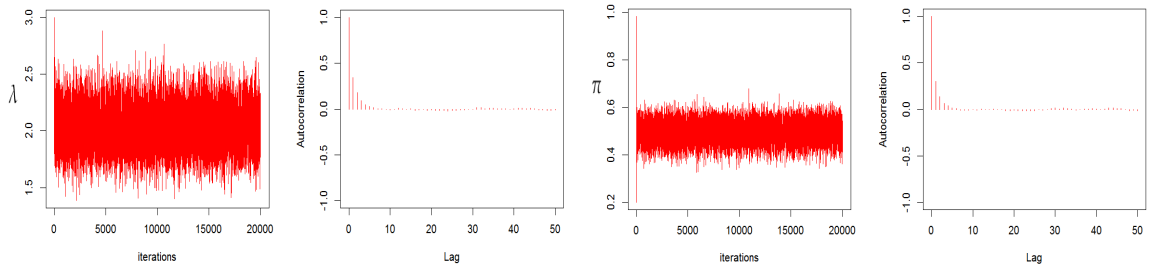
Set 2 of this simulation studies considers when the component means are far apart. The first set of true parameters are $(\lambda, \pi) = (4, 0.2)$ and $(r_2, v_2) = (20, 0.2)$ for the two components. The means are 3.2 and 80.0, for noise and signal components respectively. Again, another set of true parameters are $(\lambda, \pi) = (7, 0.3)$ and $(r_2, v_2) = (55, 0.4)$ and the means respectively are $(4.9, 82.5)$. Finally the true values for the last case are $(\lambda, \pi) = (2, 0.5)$ and $(r_2, v_2) = (75, 0.6)$ with the means of the components as $(1.0, 50.0)$ respectively.

The observations are modelled using the MRF model with ZIP distribution and the non-parametric distribution. We sample 20,000 MCMC steps with 10,000 as burn-in steps, repeated 100 times. From Table 6.4, it could be seen that the estimates for the true parameters are precise. We also have a smaller classification errors.

Figure 6.2 shows trace plots and autocorrelation plots for the last set of parameter values in simulation Set 2. Each row of Figure 6.2 corresponds to two parameters, so there are two plots for each parameter. The left plot for a parameter is a trace plot - it shows the values the parameter took during the runtime of the chain. The right plot is the autocorrelation plot - values of autocorrelation against lag t (where lag t is the correlation between $g(\theta^{(s)})$ and $g(\theta^{(s+t)})$ - elements that are t time steps apart). If the chain is mixing adequately, the value of the autocorrelation decreases to zero as the tag value increases (Albert 2009). Figure 6.2 show the convergence of the Markov chain to the stationary distribution. In each of the autocorrelation plots on Figure 6.2, the value of the autocorrelation decreases to zero as the tag value increases, which indicates that the chain is mixing adequately.

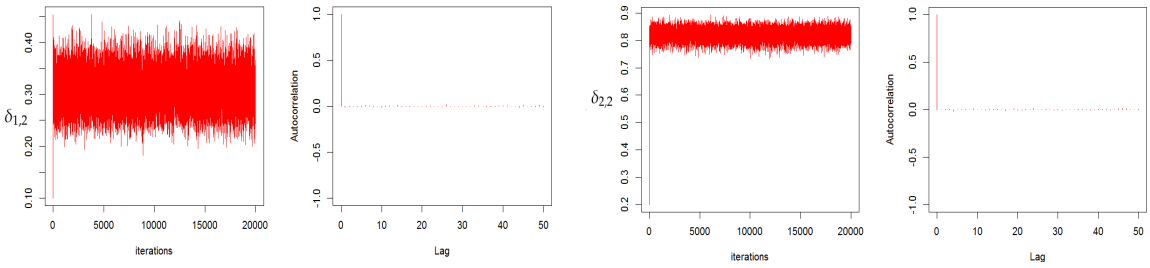
Table 6.4: Set 2 simulation results (posterior means and 95% credible intervals) where the true model is a Markov mixture model of ZIP and NB distributions.

True value				Posterior mean			Error rate
λ	π	r_2	v_2	$E(\lambda)$	$E(\pi)$	$E(\delta_{1,2})$	$E(\delta_{2,2})$	e
4	0.2	20	0.2	4.0004	0.1820	0.3212	0.8231	0.009
				(3.6641,4.3480)	(0.1271,0.2432)	(0.2521,0.3954)	(0.7790,0.8634)	
7	0.3	55	0.4	6.8380	0.2912	0.3204	0.8247	0.01
				(6.3798,7.3102)	(0.2277,0.3590)	(0.2512,0.3949)	(0.7808,0.8648)	
2	0.5	75	0.6	2.0266	0.4956	0.3097	0.8224	0.006
				(1.6767,2.3965)	(0.4078,0.5803)	(0.2422,0.3820)	(0.7786,0.8624)	



(a) True value $\lambda = 2$

(b) True value $\pi = 0.5$



(c) True value $\delta_{1,2} = 0.3$

(d) True value $\delta_{2,2} = 0.8$

Figure 6.2: Trace and autocorrelation plots for simulation in set 2 for true parameters $\lambda = 2$, $\pi = 0.5$, $\delta_{1,2} = 0.3$ and $\delta_{2,2} = 0.8$ where the true model is a Markov mixture model of ZIP and NB distributions.

6.3.3 Scenario 3

In this last Scenario, we generate $n = 500$ observations from the true Markov mixture model of Zero-inflated NB distribution and NB distribution, where $\delta_0 = (0.1, 0.9)$, $\delta_{1,2} = 0.3$ and $\delta_{2,2} = 0.8$.

As in the previous scenarios, the simulation represents two sets; when the means of the two components are not too far apart, and when the component means are far apart. In set 1 of the simulation, we chose the following true parameters; $(r_1, v_1, \pi) = (6, 0.4, 0.4)$ for the first component, and $(r_2, v_2) = (8, 0.2)$ for the second component. These give the respective means of $(5.4, 32)$. We again chose true parameter values for the two components as $(r_1, v_1, \pi) = (5, 0.5, 0.3)$ and $(r_2, v_2) = (7, 0.2)$, and the corresponding means

for the two components are (3.5, 28). And finally, we consider true parameter values for two components as $(r_1, v_1, \pi) = (3, 0.2, 0.5)$ and $(r_2, v_2) = (37, 0.6)$, which give the corresponding means as (6, 24.67). The results of the analysis are shown on Table 6.5.

Table 6.5: Set 1 simulation results (posterior means and 95% credible intervals) where the true model is a Markov mixture model of ZINB and NB distributions.

True value				Posterior mean				Error rate
r_1	r_2	v_1	v_2	π	$E(r_1)$	$E(v_1)$	$E(\pi)$	$E(\delta_{1,2})$	$E(\delta_{2,2})$	e
6	8	0.4	0.2	0.4	5.9595	0.4389	0.3565	0.3930	0.7449	0.09
					(3.7747,9.0510)	(0.3615,0.5126)	(0.2820,0.4376)	(0.3161,0.4744)	(0.6561,0.8157)	
5	7	0.5	0.2	0.3	4.9454	0.4334	0.2762	0.4021	0.7332	0.11
					(2.9206,8.2085)	(0.3484,0.5190)	(0.2067,0.3485)	(0.3234,0.4860)	(0.6324,0.8015)	
3	37	0.2	0.6	0.5	3.2171	0.4425	0.4983	0.3405	0.8054	0.03
					(1.7119,6.5786)	(0.3316,0.5469)	(0.4132,0.5844)	(0.2663,0.4202)	(0.7402,0.8531)	

Table 6.6 shows simulation results for set 2 when the component means are clearly separated. First, we consider the true parameter values $(r_1, v_1, \pi) = (3, 0.5, 0.3)$ for the first component and $(r_2, v_2) = (25, 0.2)$ for the second component, which give the corresponding means for the two components as (2.1, 100). Again we chose the true values for the two components as $(r_1, v_1, \pi) = (7, 0.5, 0.7)$ and $(r_2, v_2) = (45, 0.4)$. The corresponding means for the two components are (2.1, 67.5). Finally, the true parameters for the last set are $(r_1, v_1, \pi) = (5, 0.6, 0.5)$ and $(r_2, v_2) = (35, 0.2)$. The means for the two components are respectively (1.67, 140). Such choices, as stated earlier, give the means for the two components clearly separated, with the noise component having smaller mean value

The above observations are modelled using the MRF model with ZINB distribution and the non-parametric distribution. This is based on 20,000 MCMC iterations with 10,000 as burn-in steps, repeated for 100 times. The results of these simulations also give clear estimates, and return smaller classification errors (see Table 6.6).

Table 6.6: Set 2 simulation results (posterior means and 95% credible intervals) where the true model is a Markov mixture model of ZINB and NB distributions.

True value		Posterior mean				Error rate
$r_1\ r_2\ v_1\ v_2\ \pi$	$E(r_1)$	$E(v_1)$	$E(\pi)$	$E(\delta_{1,2})$	$E(\delta_{2,2})$	e
3 250.50.20.3	4.0310	0.5730	0.3084	0.3220	0.8229	0.01
	(2.1144,7.3757)	(0.4360,0.7063)	(0.2161,0.3985)	(0.2538,0.3961)	(0.7784,0.8637)	
7 450.50.40.7	7.5537	0.5029	0.7423	0.3415	0.8190	0.02
	(4.3211,12.8086)	(0.3928,0.6119)	(0.6707,0.8088)	(0.2678,0.4202)	(0.7744,0.8603)	
5 350.60.20.5	5.0258	0.4858	0.5349	0.3426	0.8178	0.02
	(2.8026,8.5564)	(0.3751,0.6079)	(0.4579,0.6105)	(0.2652,0.4266)	(0.7725,0.8590)	

Finally, the simulation plots for the observations for all the scenarios are displayed in Appendix B.

6.4 Data analysis

In this section we analyse ChIP-seq data set obtained from the R package `enRich` (Bao & Vinciotti 2013) generated by Ramos et al. (2010) for the detection of histone acetyltransferases for a single ChIP-seq experiment. The data set, p300T301, is one of the two technical replicates of p300, a transcriptional activator. The data set considered is only for chromosome21 and is fully described in Chapter 3 and the reference therein.

6.4.1 The proposed method

Available literature in ChIP-seq studies allow parametric distributions for the noise component to follow either Poisson distribution, NB distribution or zero-inflated distributions (see for example, Qin et al. (2010), Bao et al. (2013), Spyrou et al. (2009), Kuan et al. (2011) and Bao et al. (2014)). The fit for Poisson model is always inferior to NB model (Bao et al. 2013). Here, we show the application of the proposed MRF model with NB or zero-

inflated distributions (e.g. ZIP or ZINB distributions) for the noise component and the non-parametric distribution for the signal component on the ChIP-seq data.

Based on the posterior probabilities in (6.13), such that $g_i = \Pr(z_i = 1|\boldsymbol{x})$, a region i is considered enriched if $g_i < \rho$, where ρ is a threshold determined by controlling the FDR in (4.16) at a predefined level (Bao et al. 2014, Scott 2002). Here, we controlled the FDR at 0.1% and determined the threshold ρ. As in the previous Chapter, the FDR is controlled at the smallest value of 0.1% to avoid including genomic regions that exhibits unstructured and anomalous read counts from the analysis. Figure 6.3 shows the result for the proposed method. It can be seen that the overlap between the zero-inflated models for the noise component is greater than the NB model. The ZIP distribution for the noise component, furthermore, detected more enriched regions than the ZINB distribution at the same FDR as shown on Figure 6.3.

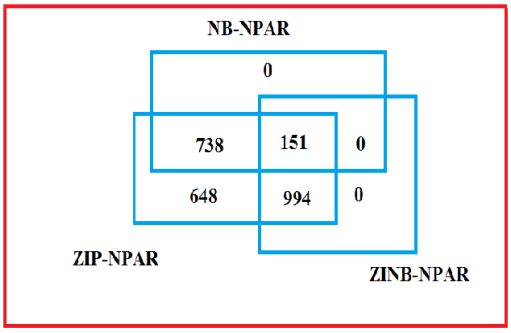


Figure 6.3: Number of enriched regions identified by the proposed methods: NB distribution for the noise component (NB-NPAR), ZIP distribution for the noise component (ZIP-NPAR), and ZINB distribution for the noise component (ZIP-NPAR) at 0.1% FDR.

6.4.2 Model comparison

For the purpose of comparison, ChIP-seq data is analysed using existing one-dimesional Markov random field models, where parametric distributions are considered for the two

components. The parametric distributions used to model the noise and the signal components are respectively: (1) ZIP distribution and Poisson distribution (ZIP-Poisson mixture), (2) ZIP and NB distribution (ZIP-NB mixture) and (3) ZINB distribution and NB distribution (ZINB-NB mixture). All the existing methods account for spatial dependencies by assuming first-order Markov property. We compared these existing methods with our proposed MRF model where the noise component follows ZIP distribution and the signal component follows the non-parametric distribution. Figure 6.4 show the enriched regions detected by the four models when FDR is controlled at 0.1%. From Figure 6.4, it can be seen how the overlap between the proposed method and ZIP-Poisson mixture and ZIP-NB mixture are both larger than the overlap between ZIP-Poisson mixture and ZIP-NB mixture at the same FDR. Furthermore, the proposed method detected more enriched regions than the existing methods.

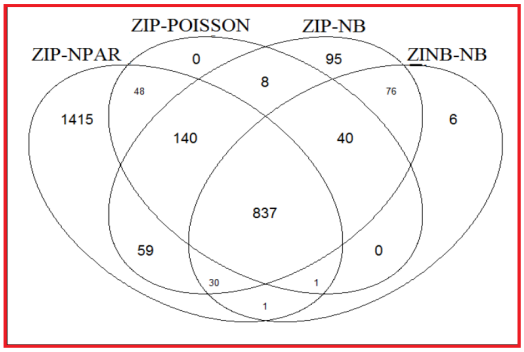


Figure 6.4: Number of enriched regions identified by the proposed method (ZIP-NPAR), mixture of ZIP and Poisson distributions (ZIP-POISSON), mixture of ZIP and NB distributions (ZIP-NB) and mixture of ZINB and NB distributions (ZINB-NB) at 0.1% FDR.

We again use the `enRich` R package (Bao & Vinciotti 2013) implementation for the existing methods. The enriched regions have the same quality as those found by the finite mixture model approach in Chapter 5.

6.5 Conclusion

We followed the existing works of Bao et al. (2014) and Mo (2012) which capture the spatial dependencies by first-order Markov property assumptions. Based on the empirical results using simulation studies, we have justified that our proposed model is sound in terms of precise estimate and classification. We demonstrated the applicability of the proposed method on real data for the detection of enriched regions for histone proteins of transcriptional activator from ChIP-seq experiment. We compared our result with existing methods, and found that our method detected more enriched regions than the existing methods at the same FDR.

CHAPTER 7

CONCLUSION AND FUTURE DIRECTION

7.1 Introduction

This research work has developed a mixture model with parametric and non-parametric components for classifying observations into noise and signal. The advantage of this new method is that it bypasses the challenges involved in the Bayesian mixture models, such as the label switching problem and the determination of the number of components K. This Chapter summarizes the contributions of the thesis, and outlines future direction.

7.2 Mixture model

The challenges in Bayesian analysis for mixture models are the label switching problem and the determination of the number of components K. The existing methods of dealing with these problems have a large computational cost, which made them unsuitable for large data sets and models with several components. Furthermore, these methods are only

suitable when the component distributions are of the same type. We proposed a mixture model approach when the interest is classifying the observations into two classes, signal and noise. For illustration, we used discrete data generated by ChIP-seq experiments, where the interest lies in whether a region of the genome is bound by the protein of interest or not. It is shown empirically, using simulation studies, that the new method can distinguish whether an observation is signal or noise, and it can do so with higher accuracy than a mixture of parametric distributions. The new method is robust to different priors as demonstrated in Appendix A. Finally, in ChIP-seq data application, where we considered 1000 contiguous base pairs regions on chromosome 21, the proposed method showed superiority when compared to similar models. We used `enRich` R package (Bao & Vinciotti 2013) implementation for the existing methods. ChromHMM (Ernst & Kellis 2010) was used to validate the enriched regions identified by the methods.

In the new proposed method, the first component h_1 could be any parametric discrete distribution of the mixture component. In ChIP-seq data for 1000 contiguous base pairs regions for example, the Poisson distribution is a natural choice. The second component h_2 is modelled as a non-parametric distribution. The non-parametric component involved L unknown parameters with p_j as the probabilities for distinct observations values in the enriched regions, interpreted as the probability of $x = x_j$ given that x is drawn from the enriched region. Here, $p_j,\ j = 1,\ldots,L$ are still unknown parameters because any non-parametric estimator uses a certain number of parameters. The number of parameters depends on the number of observations and usually much more than the number of parameters for a parametric estimator. For example, the empirical distribution uses n parameters (the jumps in the empirical distribution estimate) if there are n observations.

Our proposed signal component is non-parametric since the number of parameters $P_j,\ j = 1, \ldots, L$ will increase when the sample size n increases. Also P_j is the weight (probability) for the j^{th} ordered statistic, which will also vary for different data set. It is purely non-parametric. In this discrete example, the number of parameters is less than the sample size n, but this is because for discrete data there are ties: therefore, the method is efficient.

7.3 Markov random model

As noted earlier, the unknown parameters $p_j,\ j = 1, \ldots, L$ increase as the sample size n increases. When L becomes large, the method may not be practical due to computational cost. Overcoming this problem, in ChIP-seq data for example, is achieved by considering smaller window sizes for the genomic regions, e.g. 200 base pairs long, which is also the fragments size used in ChIP-seq experiments, thus reducing the value of L automatically. In addition to the spatial dependencies between the neighbouring windows in the data, a larger part of the genome contains an excess of zeros. This requires models that account for spatial dependencies with distributions that carter for the excess of zeros in the noise component. This is our motivation to use one-dimensional Markov random field model, with negative Binomial distribution or zero-inflated distributions (e.g. zero-inflated Poisson or zero-inflated negative Binomial distributions) for the noise component and the non-parametric distribution for the signal component.

The developed method satisfy the assumption of first-order Markov property on the latent states, which effects conditional dependencies over a long range on the observations. The empirical results, from simulations, have shown the efficiency of the method. The ap-

plications of the proposed method on ChIP-seq data, to detect enriched regions for histone proteins, using three different distributions for the noise component, are demonstrated. We found that ZIP distribution for the noise component detected more enriched regions than the other two distributions (NB distribution and ZINB distribution) at the same FDR. Furthermore, the new method outperformed similar existing methods at the same FDR. For the existing methods, `enRich` R package implementation was used. These enriched regions are of the same quality as those detected by the mixture model approach.

7.4 Contributions of the study

The non-parametric component achieved several advantages: one does not need to estimate the number of components for the mixture, neither do we need to justify distributions for the signal component, nor consider the label switching problem.

The detected gene enriched regions in a DNA sequence are of interest to Bioinformaticians. They can be use to understand epigenetic modifications during normal development and disease states. Transcription binding sites can also be use to understand cell differentiation, environmental and drug responses, and alterations to these responses during disease states.

7.5 Limitations of the proposed methods

The proposed method may not be efficient if L values become very large, since the non-parametric signal component may have n parameters, which will not be computationally efficient. Another limitation of the proposed method is that it is only valid for discrete

observations, since p_js are probabilities for the distinct values of the observations. In addition, the posterior (4.10) of z_i in Algorithm 4 will not be valid for continuous cases.

7.6 Future direction

Following the successful completion of this study, there still remain a number of potential areas for future direction. The current project is based on a two-dimensional DNA sequence, but in reality the long DNA chain is folded and genes could interact even if they are far away. One can use this methodology on a three-dimensional sequencing data, which have the information of long-range interaction between genes. The future direction is to use our proposed model, but the latent variable should be extended to several hidden layers. This will deal with such genetic problems.

Our proposed method can be used in other application areas for classification. For example, in signal processing applications, where the interest lies in distinguishing a speech signal from a corrupted noise and transmission distortion, our model can provide an avenue that will optimally process the signal to achieve a desired output. In machine learning, developing a reliable feature selection method is an active research area. Another potential application area of our methodology in machine learning, therefore, is that of classifying informative and non-informative features for discrete data. It will explicitly assist the learner to focus on relevant features and ignore irrelevant ones prior to learning.

Finally, another future research direction is to improve on the limitations of the proposed method. This is to allow the non-parametric signal component to have n parameters (just like the empirical likelihood). The possibility here is to consider a kernel density fitting.

Such method can deal with the continuous cases.

BIBLIOGRAPHY

Albert, J. (2009), *Bayesian computation with R*, Springer Science & Business Media.

Antoniak, C. E. (1974), 'Mixtures of dirichlet processes with applications to bayesian non-parametric problems', *The Annals of Statistics* **2**, 1152–1174.

Bao, Y. & Vinciotti, V. (2013), *An R package for the analysis of multiple ChIP-seq data*. R package version 2.0.

Bao, Y., Vinciotti, V., Wit, E. & ACt Hoen, P. (2013), 'Accounting for immunoprecipitation efficiencies in the statistical analysis of chip-seq data', *BMC Bioinformatics* **14**(1), 169.

Bao, Y., Vinciotti, V., Wit, E. & AC't Hoen, P. (2014), 'Joint modeling of chip-seq data via a markov random field model', *Biostatistics* **15**(2), 296–310.

Beckett, S., Jee, J., Ncube, T., Pompilus, S., Washington, Q., Singh, A. & Pal, N. (2014), 'Zero-inflated poisson (zip) distribution: Parameter estimation and applications to model data from natural calamities', *Journal of Mathematics* **7**(6), 751–767.

Bernardo, J. M. & Smith, A. F. (2009), *Bayesian theory*, Vol. 405, John Wiley & Sons.

Blake, A., Kohli, P. & Rother, C. (2011), *Markov random fields for vision and image processing*, Mit Press.

Bolstad, W. M. (2011), *Understanding computational bayesian statistics*, Vol. 644, John Wiley & Sons.

Bradlow, E. T., Hardie, B. G. & Fader, P. S. (2002), 'Bayesian inference for the negative binomial distribution via polynomial expansions', *Journal of Computational and Graphical Statistics* **11**, 189–202.

Bremaud, P. (1999), *Markov chains: Gibbs fields, monte carlo simulation, and queues*, Vol. 31, Springer Science & Business Media.

Carlin, B. P. & Louis, T. A. (2011), *Bayesian methods for data analysis*, CRC Press.

Casella, G. & George, E. I. (1992), 'Explaining the gibbs sampler', *The American Statistician* **46**(3), 167–174.

Celeux, G. (1998), Bayesian inference for mixture: The label switching problem, *in* R. Payne & P. Green, eds, 'COMPSTAT 1998', Heidelberg and Vienna: Physica-Verlag, pp. 227–232.

Celeux, G., Hurn, M. & Robert, C. P. (2000), 'Computational and inferential difficulties with mixture posterior distributions', *Journal of the American Statistical Association* **95**(451), 957–970.

Chandgotia, N., Han, G., Marcus, B., Meyerovitch, T. & Pavlov, R. (2014), One-dimensional

markov random fields, markov chains and topological markov fields, *in* 'Proceeding of the American Mathematical Society', Vol. 142, pp. 227–242.

Chaudhary, S. (2014), 'Implementation and performance analysis of markov random field', *International Journal of Advancements in Research and Technology* **3**(1), 37–41.

Chen, M.-H., Shao, Q.-M. & Ibrahim, J. G. (2012), *Monte carlo methods in bayesian computation*, Springer Science & Business Media.

Chen, W., Jia, W., Wang, K., Si, X., Zhu, S., Duan, T. & Kang, J. (2013), 'Distinct roles for cbp and p300 on the ra-mediated expression of the meiosis commitment gene stra8 in mouse embryonic stem cells', *PLoS One* **8**(6), e66076.

Chib, S. & Greenberg, E. (1995), 'Understanding the metropolis-hastings algorithm', *The American Statistician* **49**(4), 327–335.

Cook, J. D. (2009), 'Notes on the negative binomial distribution', `http://www.johndcook.com/negative_binomial.pdf`. Accessed: 2015-12-09.

Diebolt, J. & Robert, C. P. (1994), 'Estimation of finite mixture distributions through bayesian sampling', *Journal of the Royal Statistical Society. Series B* (56), 363–375.

Ernst, J. & Kellis, M. (2010), 'Discovery and characterization of chromatin states for systematic annotation of the human genome', *Nature Biotechnology* **28**(8), 817–825.

Escobar, M. D. & West, M. (1995), 'Bayesian density estimation and inference using mixtures', *Journal of the American Statistical Association* **90**(430), 577–588.

Ferguson, T. S. (1973), 'A bayesian analysis of some nonparametric problems', *The Annals of Statistics* **1**, 209–230.

Flynn, M. & Francis, L. A. (2009), More flexible glms zero-inflated models and hybrid models, *in* 'Proceeding of Casualty Actuarial Society E-Forum, Winter 2009', pp. 148–224.

Frühwirth-Schnatter, S. (2006), *Finite mixture and markov switching models: Modeling and applications to random processes*, Springer Science & Business Media.

Gelman, A. (2002), 'Prior distribution', *Encyclopedia of Environmetrics* **3**, 1634–1637.

Gelman, A., Carlin, J. B., Stern, H. S. & Rubin, D. B. (2014), *Bayesian data analysis*, Vol. 2, Taylor & Francis.

Geman, S. & Geman, D. (1984), 'Stochastic relaxation, gibbs distributions, and the bayesian restoration of images', *IEEE Transactions on Pattern Analysis and Machine Intelligence* (6), 721–741.

Geweke, J., Gowrisankaran, G. & Town, R. J. (2003), 'Bayesian inference for hospital quality in a selection model', *Econometrica* **71**(4), 1215–1238.

Ghahramani, Z. (2001), 'An introduction to hidden markov models and bayesian networks', *International Journal of Pattern Recognition and Artificial Intelligence* **15**(1), 9–42.

Gilks, W. R., Richardson, S. & Spiegelhalter, D. J. (1996), Introducing markov chain monte carlo, *in* W. R. Gilks, S. Richardson & D. J. Spiegelharter, eds, 'Markov Chain Monte Carlo in Practice', Vol. 1, Chapman and Hall, London, pp. 1–19.

Green, P. J. (1995), 'Reversible jump markov chain monte carlo computation and bayesian model determination', *Biometrika* **82**(4), 711–732.

Guttorp, P. & Minin, V. N. (1995), *Stochastic modeling of scientific data*, CRC Press.

Jasra, A., Holmes, C. & Stephens, D. (2005), 'Markov chain monte carlo methods and the label switching problem in bayesian mixture modeling', *Statistical Science* (20), 50–67.

Jung, K. (2009), 'Markov random field: Theory and application', `http://web.kaist.ac.kr/~kyomin/Fall09MRF/lec02.pdf`. Accessed: 2015-09-08.

Kindermann, R. & Snell, J. L. (1980), *Markov random fields and their applications*, Vol. 1, American Mathematical Society Providence.

Koski, T. (2001), *Hidden Markov models for bioinformatics*, Vol. 2, Springer Science & Business Media.

Kuan, P. F., Chung, D., Pan, G., Thomson, J. A., Stewart, R. & Keleş, S. (2011), 'A statistical framework for the analysis of chip-seq data', *Journal of the American Statistical Association* **106**(495), 891–903.

Lambert, D. (1992), 'Zero-inflated poisson regression, with an application to defects in manufacturing', *Technometrics* **34**(1), 1–14.

Lefrançois, P., Zheng, W. & Snyder, M. (2010), 'Chip-seq: Using high-throughput dna sequencing for genome-wide identification of transcription factor binding sites', *Methods in Enzymology* **470**, 77–104.

Lempitsky, V., Rother, C., Roth, S. & Blake, A. (2010), 'Fusion moves for markov random field optimization', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **32**(8), 1392–1405.

Lim, J. (2010), 'Chip-seq: The new way to seq genome wide for transcription factor binding patterns', `http://biochem218.stanford.edu/Projects%202010/Lim%202010.pdf`. Accessed: 2014-11-20.

Marin, J.-M., Mengersen, K. & Robert, C. P. (2005), Bayesian modelling and inference on mixtures of distributions, *in* D. Dey & C. Rao, eds, 'Handbook of Statistics', Vol. 25, Elsevier-Sciences, London, pp. 459–507.

Marin, J.-M. & Robert, C. P. (2014), *Bayesian essentials with R*, Springer.

McLachlan, G. & Peel, D. (2004), *Finite mixture models*, John Wiley & Sons.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. & Teller, E. (1953), 'Equation of state calculations by fast computing machines', *The Journal of Chemical Physics* **21**(6), 1087–1092.

Mo, Q. (2012), 'A fully bayesian hidden ising model for chip-seq data analysis', *Biostatistics* **13**, 113–128.

Nature Education (2014), Scitable: A collaborative learning for space and science, `http://www.nature.com/scitable`. Accessed: 2015-09-19.

Nicholl, D. S. (2008), *An introduction to genetic engineering*, Cambridge University Press.

Nix, D. A., Courdy, S. J. & Boucher, K. M. (2008), 'Empirical methods for controlling false positives and estimating confidence in chip-seq peaks', *BMC Bioinformatics* **9**(1), 523.

Nobile, A. & Fearnside, A. T. (2007), 'Bayesian finite mixtures with an unknown number of components: The allocation sampler', *Statistics and Computing* **17**(2), 147–162.

Park, P. J. (2009), 'Chip–seq: Advantages and challenges of a maturing technology', *Nature Reviews Genetics* **10**(10), 669–680.

Pearson, K. (1894), 'Contributions to the mathematical theory of evolution', *Philosophical Transactions of the Royal Society of London, A* **185**, 71–110.

Press, S. J. (2009), *Subjective and objective bayesian statistics: principles, models, and applications*, Vol. 590, John Wiley & Sons.

Qin, Z. S., Yu, J., Shen, J., Maher, C. A., Hu, M., Kalyana-Sundaram, S., Yu, J. & Chinnaiyan, A. M. (2010), 'Hpeak: An hmm-based algorithm for defining read-enriched regions in chip-seq data', *BMC Bioinformatics* **11**(1), 369.

Rabiner, L. (1989), A tutorial on hidden markov models and selected applications in speech recognition, *in* 'Proceedings of the IEEE', Vol. 77, pp. 257–286.

Ramos, Y. F., Hestand, M. S., Verlaan, M., Krabbendam, E., Ariyurek, Y., van Galen, M., van Dam, H., van Ommen, G.-J. B., den Dunnen, J. T., Zantema, A. & ACt Hoen, P. (2010), 'Genome-wide assessment of differential roles for p300 and cbp in transcription regulation', *Nucleic Acids Research* (38), 5396–5408.

Richardson, S. & Green, P. J. (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.

Robert, C. & Casella, G. (2009), *Introducing monte carlo methods with R*, Springer Science & Business Media.

Robert, C. P., Ryden, T. & Titterington, D. M. (2000), 'Bayesian inference in hidden markov models through the reversible jump markov chain monte carlo method', *Journal of the Royal Statistical Society. Series B* **62**(1), 57–75.

Rodríguez, C. E. & Walker, S. G. (2014), 'Label switching in bayesian mixture models: Deterministic relabeling strategies', *Journal of Computational and Graphical Statistics* **23**(1), 25–45.

Salakhutdinov, R. R. (2009), Learning in markov random fields using tempered transitions, *in* Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams & A. Culotta, eds, 'Advances in Neural Information Processing Systems', MIT, press, pp. 1598–1606.

Schweikert, G., Cseke, B., Clouaire, T., Bird, A. & Sanguinetti, G. (2013), 'Mmdiff: Quantitative testing for shape changes in chip-seq data sets', *BMC genomics* **14**(1), 826.

Scott, S. L. (2002), 'Bayesian methods for hidden markov models: Recursive computing in the 21st century', *Journal of the American Statistical Association* **97**(457), 337–351.

Sheskin, D. J. (2003), *Handbook of parametric and nonparametric statistical procedures*, crc Press.

Song, S., Nicolae, D. L. & Song, J. (2010), 'Estimating the mixing proportion in a semiparametric mixture model', *Computational Statistics & Data Analysis* **54**(10), 2276–2283.

Sperrin, M., Jaki, T. & Wit, E. (2010), 'Probabilistic relabelling strategies for the label switching problem in bayesian mixture models', *Statistics and Computing* **20**(3), 357–366.

Spyrou, C., Stark, R., Lynch, A. G. & Tavaré, S. (2009), 'Bayespeak: Bayesian analysis of chip-seq data', *BMC Bioinformatics* **10**(1), 299.

Stephens, M. (2000*a*), 'Bayesian analysis of mixture models with an unknown number of components-an alternative to reversible jump methods', *Annals of Statistics* **28**, 40–74.

Stephens, M. (2000*b*), 'Dealing with label switching in mixture models', *Journal of the Royal Statistical Society. Series B* **62**(4), 795–809.

Suganuma, T. & Workman, J. L. (2011), 'Signals and combinatorial functions of histone modifications', *Annual review of biochemistry* **80**, 473–499.

Turchin, V. F. (1971), 'On the computation of multidimensional integrals by the monte-carlo method', *Theory of Probability & Its Applications* **16**(4), 720–724.

U.S. National Library of Medicine (2015), Genetics home reference handbook: Help me understand genetics, `https://ghr.nlm.nih.gov/primer.pdf`. Accessed: 2015-03-15.

Venturini, S., Dominici, F. & Parmigiani, G. (2008), 'Gamma shape mixtures for heavy-tailed distributions', *The Annals of Applied Statistics* **2**, 756–776.

Vinciotti, V. & Bao, Y. (2013), 'Discovery of protein binding patterns by joint modelling of chip-seq data', `http://www2.mate.polimi.it/convegni/viewpaper.php?id=365&print=1&cf=33`. Accessed: 2015-10-02.

Wang, C., Komodakis, N. & Paragios, N. (2013), 'Markov random field modeling, inference & learning in computer vision & image understanding: A survey', *Computer Vision and Image Understanding* **117**(11), 1610–1627.

Wang, J., Huda, A., Lunyak, V. V. & Jordan, I. K. (2010), 'A gibbs sampling strategy applied to the mapping of ambiguous short-sequence tags', *Bioinformatics* **26**(20), 2501–2508.

Wang, Z., Zang, C., Cui, K., Schones, D. E., Barski, A., Peng, W. & Zhao, K. (2009), 'Genome-wide mapping of hats and hdacs reveals distinct functions in active and inactive genes', *Cell* **138**(5), 1019–1031.

Wei, Z. & Li, H. (2007), 'A markov random field model for network-based analysis of genomic data', *Bioinformatics* **23**(12), 1537–1544.

West, M. (1997), 'Hierarchical mixture models in neurological transmission analysis', *Journal of the American Statistical Association* **92**(438), 587–606.

Williams, C. (2008), 'Pmr lectures: Bayesian parameter estimation', `http://www.inf.ed.ac.uk/teaching/courses/pmr/slides/bayespe-2x2.pdf`. Accessed: 2014-09-11.

Xiang, S., Yao, W. & Wu, J. (2014), 'Minimum profile hellinger distance estimation for a semiparametric mixture model', *Canadian Journal of Statistics* **42**(2), 246–267.

Xie, F. & Chen, Z. (2012), 'Label switch in mixture model and relabeling algorithm: Project for reading course', `http://probability.ca/jeff/ftpdir/MCMCRelablelProject.pdf`. Accessed: 2016-04-11.

Yildirim, I. (2012), 'Bayesian inference: Metropolis-hastings sampling',

`http://www.bcs.rochester.edu/people/robbie/jacobslab/cheat-sheet/MetropolisHastingsSampling.pdf`. Accessed: 2014-11-20.

Zhang, Y., Brady, M. & Smith, S. (2001), 'Segmentation of brain mr images through a hidden markov random field model and the expectation-maximization algorithm', *IEEE Transactions on Medical Imaging* **20**(1), 45–57.

Zhang, Y., Liu, T., Meyer, C. A., Eeckhoute, J., Johnson, D. S., Bernstein, B. E., Nusbaum, C., Myers, R. M., Brown, M., Li, W. et al. (2008), 'Model-based analysis of chip-seq (macs)', *Genome Biology* **9**(9), R137.

APPENDIX A

PRIOR SENSITIVITY ANALYSIS AND DATA ANALYSIS TRACE PLOTS FOR THE PROPOSED METHOD

A.1 Prior sensitivity analysis

We show the sensitivity analysis of our model to the priors used. We demonstrate that the model is robust to different priors. We draw observations of size 500 from two-component mixture of Poisson and Negative Binomial distributions. The true model is;

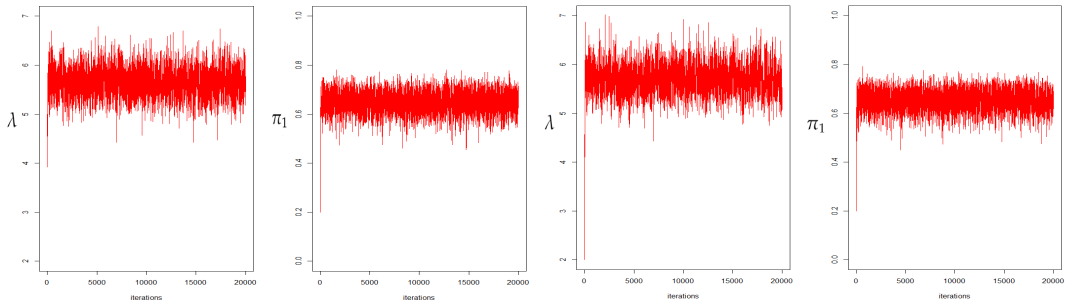
$$f(x) = \pi_1 \text{Poi}(x; \lambda) + \pi_2 \text{NB}(x; r, v) \tag{A.1}$$

where λ is the mean of the Poisson distribution, r is the nonnegative dispersion parameter and v is the probability parameter for the Negative Binomial distribution. We choose different priors and run traditional Gibbs sampling algorithm for 20,000 samples with 10,000 as burn-in iterations. We consider when $\pi_1 = 0.8$ and $\pi_1 = 0.5$. The results are presented in Tables(A.1 and A.2) and Figures (A.1 and A.2). We can see that the estimates

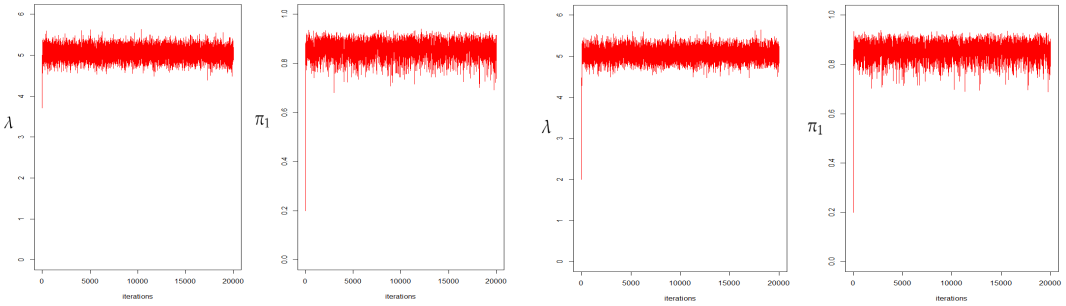
are robust under different priors. In all simulation studies there is no label switching problem.

Table A.1: Simulation results under different priors with true value of $\lambda = 5$, $r = 3$ and $v = 0.2$, where the true model is a two-component mixture distributions

Prior	True $\pi_1 = 0.8$		True $\pi_1 = 0.5$	
	$E(\lambda)$	$E(\pi_1)$	$E(\lambda)$	$E(\pi_1)$
$\alpha = 2, \beta = 1$	5.0699	0.8555	5.7305	0.6505
	(4.7802,5.3536)	(0.7876,0.9061)	(5.1844,6.313)	(0.5643,0.723)
$\alpha = 1, \beta = 2$	5.0431	0.8554	5.6367	0.6491
	(4.760,5.3244)	(0.784,0.9073)	(5.1126,6.1895)	(0.5645,0.7219)
$\alpha = 4, \beta = 2$	5.0546	0.8543	5.6994	0.6519
	(4.7703,5.3355)	(0.7838,0.9057)	(5.1595,6.2788)	(0.5661,0.7244)
$\alpha = 2, \beta = 4$	5.0051	0.8536	5.5077	0.6452
	(4.706,5.2853)	(0.715,0.9063)	(4.9652,6.0488)	(0.5572,0.7181)



(a) True value: $\pi_1 = 0.5, \alpha = 2, \beta = 1$ (b) True value: $\pi_1 = 0.5, \alpha = 4, \beta = 2$

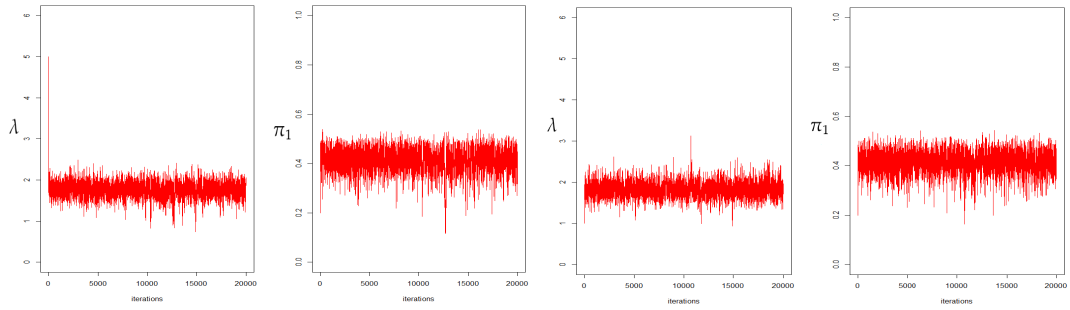


(c) True value: $\pi_1 = 0.8, \alpha = 2, \beta = 1$ (d) True value: $\pi_1 = 0.8, \alpha = 4, \beta = 2$

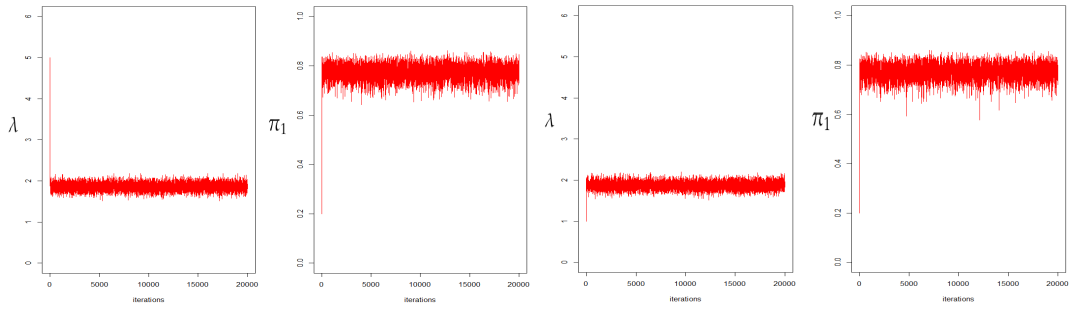
Figure A.1: Prior sensitivity plots for π_1 with true value $\lambda = 5, r = 3, v = 0.2$ where the true model is a two-component mixture distributions.

Table A.2: Simulation results for the prior sensitivity with true value of $\lambda = 2$, $r = 15$ and $v = 0.5$, where the true model is a two-component mixture distributions

Prior	True $\pi_1 = 0.8$		True $\pi_1 = 0.5$	
	$E(\lambda)$	$E(\pi_1)$	$E(\lambda)$	$E(\pi_1)$
$\alpha = 2, \beta = 1$	1.867	0.772	1.8184	0.4112
	(1.694,0.8239)	(0.7079,0.8239)	(1.4753,2.1851)	(0.3123,0.4837)
$\alpha = 1, \beta = 2$	1.8597	0.7734	1.7515	0.4086
	(1.6869,2.031)	(0.7122,0.824)	(1.3659,2.0890)	(0.2962,0.4832)
$\alpha = 4, \beta = 2$	1.8683	0.7722	1.8229	0.4117
	(1.6960,2.0416)	(0.7093,0.8235)	(1.4889,2.1851)	(0.3126,0.4841)
$\alpha = 2, \beta = 4$	1.8441	0.7718	1.728	0.410
	(1.6751,2.0139)	(0.7082,0.8228)	(1.3646,2.063)	(0.3169,0.480)



(a) True value: $\pi_1 = 0.5, \alpha = 1, \beta = 2$ (b) True value: $\pi_1 = 0.5, \alpha = 4, \beta = 2$



(c) True value: $\pi_1 = 0.8, \alpha = 1, \beta = 2$ (d) True value: $\pi_1 = 0.8, \alpha = 4, \beta = 2$

Figure A.2: Prior sensitivity plots for π_1, with true value $\lambda = 2, r = 15, v = 0.5$ where the true model is a two-component mixture distributions.

A.2 ChIP-sequence data plot

Figures A.3 and A.4 show the trace plots for the ChIP-seq data sets analysed using our proposed method for p300T301.1000bp amd CBPT301.1000bp respectively.

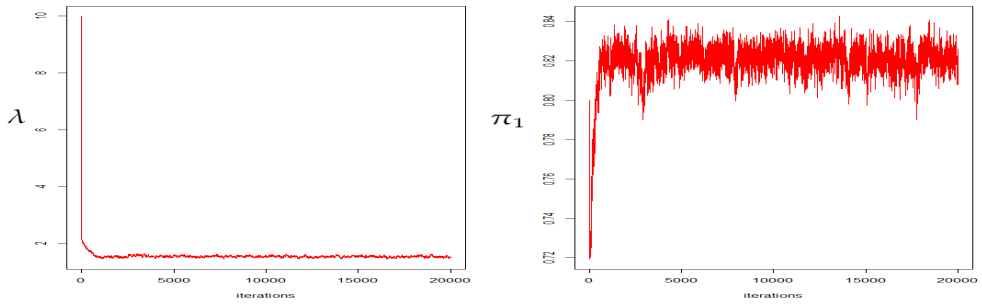


Figure A.3: Trace plots for the ChIP-sequence data (p300T301.1000bp) for chromosome21 for our proposed method for parameters λ and π_1

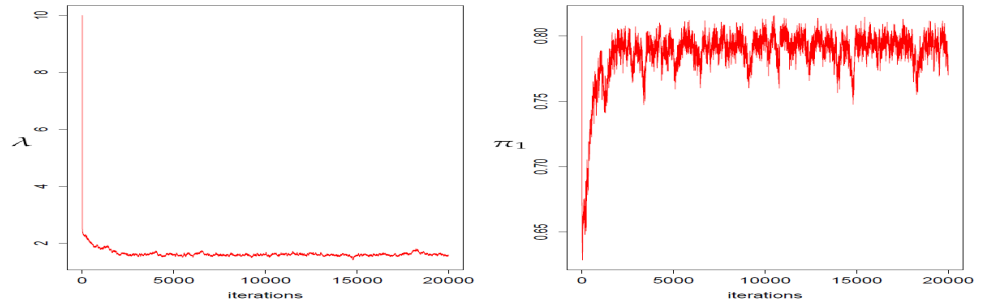


Figure A.4: Trace plots for the ChIP-sequence data (CBPT301.1000bp) for chromosome21 for our proposed method for parameters λ and π_1

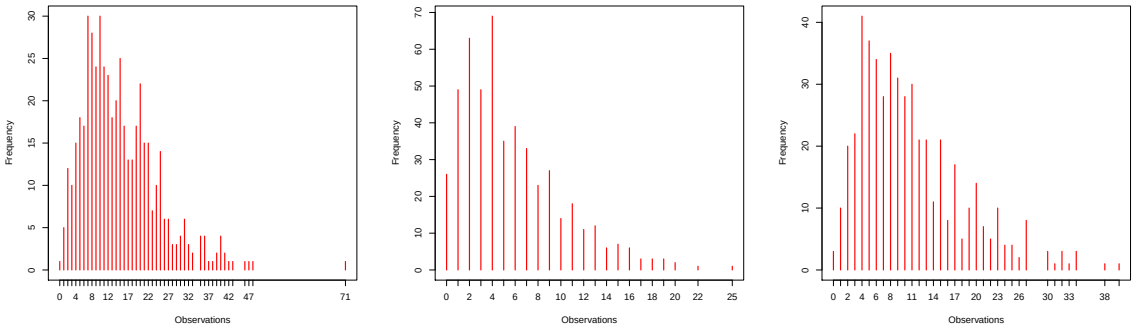
APPENDIX B

SIMULATION PLOTS FOR ONE DIMENSIONAL MARKOV RANDOM FIELD MODEL

B.1 Simulation plots

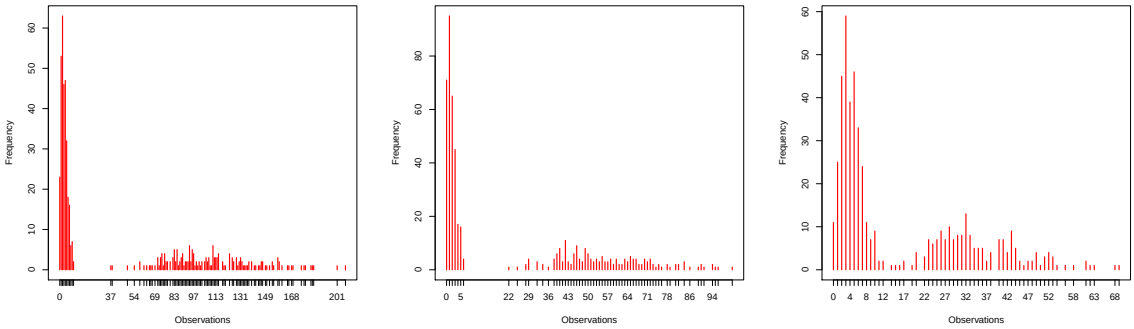
We demonstrate graphically the observations generated in Section 6.3 for the three scenarios.

B.1.1 Scenario 1



(a) (b) (c)

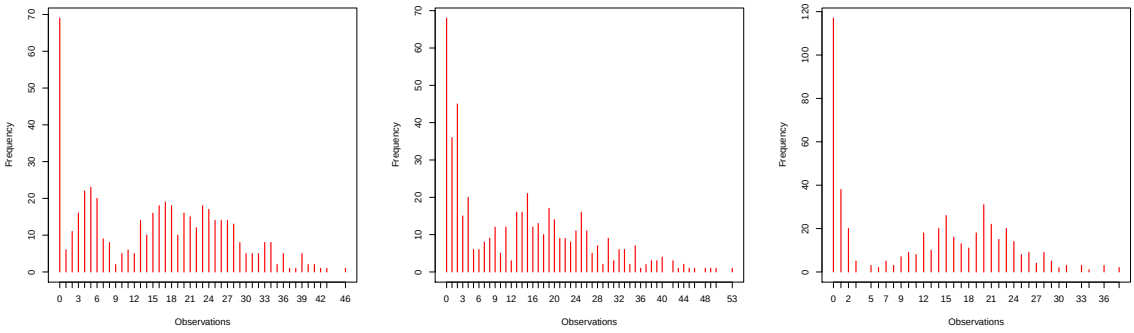
Figure B.1: Set 1 simulation plots for the true model of two-component Markov mixture model of NB distributions, where $\delta_{1,2} = 0.2$ and $\delta_{2,2} = 0.7$ for (a) $(r_1, v_1) = (3, 0.2)$, $(r_2, v_2) = (5, 0.2)$, (b) $(r_1, v_1) = (5, 0.6)$, $(r_2, v_2) = (10, 0.5)$ and (c) $(r_1, v_1) = (5, 0.4)$, $(r_2, v_2) = (7, 0.3)$.



(a) (b) (c)

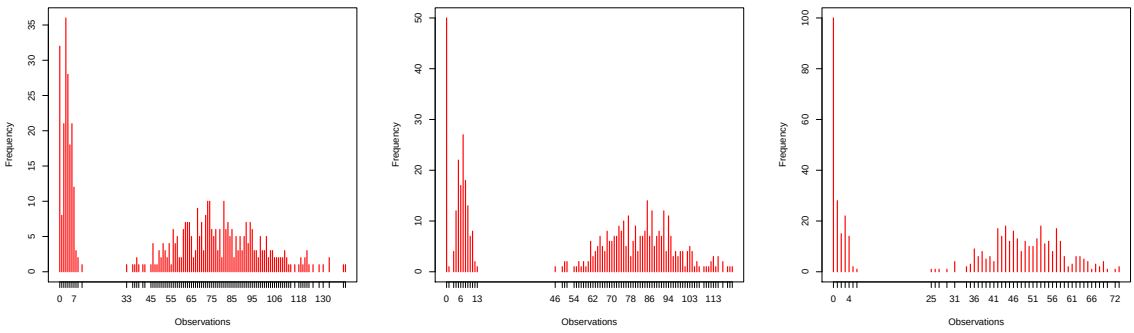
Figure B.2: Set 2 simulation plots for the true model of two-component Markov mixture model of NB distributions, where $\delta_{1,2} = 0.2$ and $\delta_{2,2} = 0.7$ for (a) $(r_1, v_1) = (5, 0.6)$, $(r_2, v_2) = (12, 0.1)$, (b) $(r_1, v_1) = (7, 0.8)$, $(r_2, v_2) = (14, 0.2)$ and (c) $(r_1, v_1) = (10, 0.7)$, $(r_2, v_2) = (15, 0.3)$.

B.1.2 Scenario 2



(a) (b) (c)

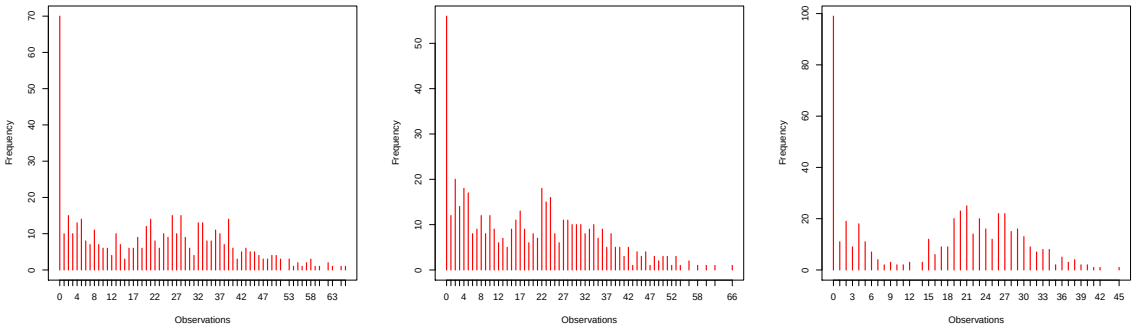
Figure B.3: Set 1 simulation plots for the true model of two-component Markov mixture model of ZIP and NB distributions, where $\delta_{1,2} = 0.3$ and $\delta_{2,2} = 0.8$ for (a) $(\lambda, \pi) = (5, 0.4)$, $(r_2, v_2) = (15, 0.4)$, (b) $(\lambda, \pi) = (2, 0.3)$, $(r_2, v_2) = (5, 0.2)$ and (c) $(\lambda, \pi) = (1, 0.5)$, $(r_2, v_2) = (19, 0.5)$.



(a) (b) (c)

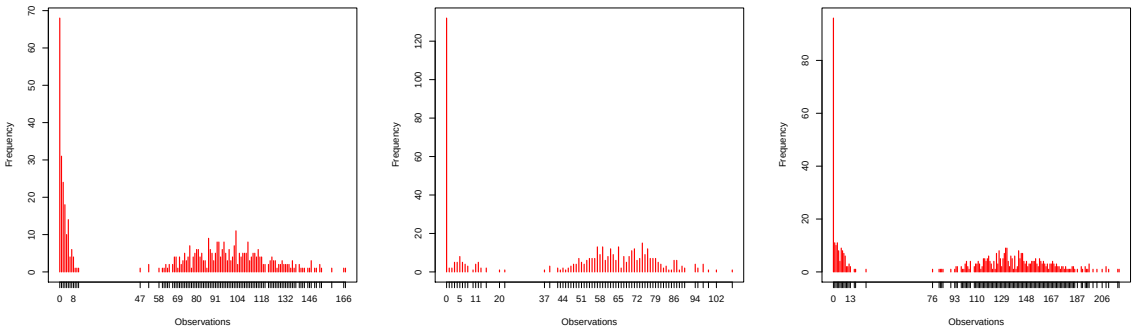
Figure B.4: Set 2 simulation plots for the true model of two-component Markov mixture model of ZIP and NB distributions, where $\delta_{1,2} = 0.3$ and $\delta_{2,2} = 0.8$ for (a) $(\lambda, \pi) = (4, 0.2)$, $(r_2, v_2) = (20, 0.2)$, (b) $(\lambda, \pi) = (7, 0.3)$, $(r_2, v_2) = (55, 0.4)$, and (c) $(\lambda, \pi) = (2, 0.5)$ and $(r_2, v_2) = (75, 0.6)$.

B.1.3 Scenario 3



(a) (b) (c)

Figure B.5: Set 1 simulation plots for the true model of two-component Markov mixture model of ZINB and NB distributions, where $\delta_{1,2} = 0.3$ and $\delta_{2,2} = 0.8$ for (a) $(r_1, v_1, \pi) = (6, 0.4, 0.4)$, $(r_2, v_2) = (8, 0.2)$, (b) $(r_1, v_1, \pi) = (5, 0.5, 0.3)$, $(r_2, v_2) = (7, 0.2)$ and (c) $(r_1, v_1, \pi) = (3, 0.2, 0.5)$ and $(r_2, v_2) = (37, 0.6)$.



(a) (b) (c)

Figure B.6: Set 2 simulation plots for the true model of two-component Markov mixture model of ZINB and NB distributions, where $\delta_{1,2} = 0.3$ and $\delta_{2,2} = 0.8$ for (a) $(r_1, v_1, \pi) = (3, 0.5, 0.3)$, $(r_2, v_2) = (25, 0.2)$, (b) $(r_1, v_1, \pi) = (7, 0.5, 0.7)$, $(r_2, v_2) = (45, 0.4)$ and (c) $(r_1, v_1, \pi) = (5, 0.6, 0.5)$, $(r_2, v_2) = (35, 0.2)$.

B.2 Distributions of ChIP-seq data

Table B.1: Summary statistics of ChIP-seq data for one experiment on the protein p300T301.200bp on chromosome21.

Sample size	min	max	Mean	Variance
234,721	0	145	0.328	1.278

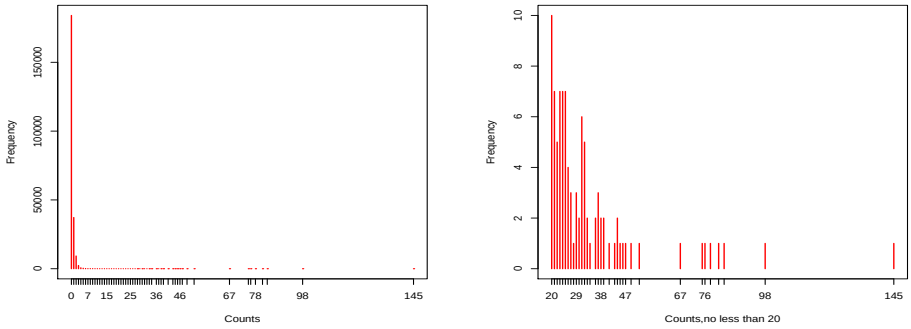


Figure B.7: Distribution of ChIP-seq data (p300T301.200bp) for one experiment (left), with zoom on the tail (right) for 200bp windows length.

APPENDIX C

THE R CODES

C.1 R code for mixture models

C.1.1 Scenario 1 (Section 4.3.1) simulation studies

C.1.1.1 The proposed method

```
rm(list=ls())
library("gtools")
library("Rlab")
generate.data <- function(truep = c(0.8,0.2),truelambda=2,v =0.4,r=15,n = 500)
  {
  x<-rep(0,n)
  truez<-rep(0,n)
  u<-runif(n)
  for(i in 1:n)
    {
      if(u[i]<truep[1]){
        x[i]=rpois(1,truelambda)
        truez[i]=0
        }
        else
        {
        x[i]=rnbinom(1,r,v)
        truez[i]=1
          }
```

```
      }
  data<-list(s=truez, o=x)
  return(data)
  }
ndata <- generate.data(truep = c(0.8,0.2),truelambda=2,v =0.4,r=15,n = 500)
x <-ndata$o
truez <- ndata$s

Newrbern = function(prob)
  {
 U = runif(length(prob))
  return(1*(U<prob))
  }
 scenario1<-function(x,MCMCsteps=20000,burnin=10000,prioralpha=1,priorbeta=2)
  {
    p_start=c(0.5,0.5)
    lambda_start= 1
    z_start=1-(x<=5)
    n = length(x)
    lambda=lambda_start
    p=p_start
    z=z_start
    CHAIN_lambda=lambda
    CHAIN_p=p[1]

  for(step in 1:MCMCsteps)
      {
       print(c("iteration",step))
        step
      subxz1=subset(x,z==1)
      freqmatrix = as.matrix(subxz1)%*%matrix(1,1,n)
      freqx = rowSums(t(freqmatrix)==x)
      Nprob = unique(cbind(x,freqx+1))
      ft2 = rdirichlet(1,Nprob[,2])
      f2 = colSums(t(as.matrix(ft2))%*%matrix(1,1,n)*(t(as.matrix(x)
      %*%matrix(1,1,length(ft2)))==Nprob[,1]))

      a = p[1] * dpois(x,lambda)
      b = p[2] * f2 + rep(0,n)
      pq= a/(a+b)
      z = 1 - Newrbern(pq)

      GammaA=sum(x*(z==0))+prioralpha
      GammaB = sum(z==0) + priorbeta
```

```
        lambda = rgamma(1, GammaA+1, GammaB)
        CHAIN_lambda = c(CHAIN_lambda,lambda)

        BetaA = sum(z==0)
        BetaB = sum(z==1)
        p[1] = rbeta(1,BetaA+1, BetaB+1)
        p[2] = 1-p[1]
        CHAIN_p = c(CHAIN_p,p[1])
        }
   para.samples=as.matrix(cbind((CHAIN_lambda[(burnin+1):MCMCsteps]),(CHAIN_p[(
     burnin+1):MCMCsteps])))
   colnames(para.samples) <- c("lambda","pie")
   para=round(apply(para.samples, 2, mean), 4)
   quan <- cbind(quantile(CHAIN_lambda[(burnin+1):MCMCsteps],prob=c(0.025,0.975)
     ),quantile(CHAIN_p[(burnin+1):MCMCsteps],prob=c(0.025,0.975)))
   colnames(quan) <- c("lambda","pie")
   return(list(parameters=para.samples,means=para,quantiles=quan))
}
   scene1 <-scenario1(x,MCMCsteps=20000,burnin=10000,prioralpha=1,priorbeta=2)
   para <-scene1$means
   lambda<-scene1$parameters[,1]
   pie<-scene1$parameters[,2]
   ############## plots ################################
   par(mfrow=c(1,2))
   plot(lambda,type="l",ylab="lambda",xlab="iterations",col="red")
   plot(pie,type="l",ylab="pie",xlab="iterations",col="red")
```

C.1.1.2 Two-components mixture of Poisson and negative Binomial distributions

```
rm(list=ls())
library("gtools")
library("Rlab")
generate.data <- function(truep=c(0.8,0.2),truelambda=2,r=15,v= 0.4,n=500)
 {
  x<-rep(0,n)
  truez<-rep(0,n)
  u<-runif(n)
  for(i in 1:n){
    if(u[i]<=truep[1])
      {
      x[i]=rpois(1,truelambda)
```

```
13        truez[i]=0
14        }
15      else{
16        x[i]=rnbinom(1,r,v)
17        truez[i]=1
18        }
19     }
20  data<-list(s=truez, o=x)
21  return(data)
22  }
23 ndata <- generate.data(truep = c(0.8,0.2),truelambda=2,r =15,v=0.4,n = 500)
24 x <-ndata$o
25 truez <- ndata$s
26
27 Newrbern = function(prob)
28 {
29   U = runif(length(prob))
30   return(1*(U<prob))
31 }
32 POISNB <- function(x,MCMCsteps=20000,burnin = 10000,prioralpha=2,priorbeta=4,
       gammaprior=c(15,1))
33 {
34 p_start=c(0.8,0.2)
35 lambda_start=1
36 r_start=5
37 v_start=0.3
38 z_start=1-(x<=5)
39 Ncomponent=2
40 n =length(x)
41 lambda=lambda_start
42 p=p_start
43 z=z_start
44 rr = r_start
45 vv = v_start
46 CHAIN_lambda=lambda_start
47 CHAIN_p=p_start
48 CHAIN_r=r_start
49 CHAIN_v=v_start
50 z=z_start
51
52  for(step in 1:MCMCsteps)
53     {
54   print(c("iteration",step))
55   step
```

```
56
57  GammaA=sum(x*(z==0))+prioralpha
58  GammaB = sum(z==0) + priorbeta
59  lambda = rgamma(1, GammaA+1, GammaB)
60  CHAIN_lambda = c(CHAIN_lambda,lambda)
61
62  Beta=NULL
63  for(k in 1:Ncomponent){
64    Beta = c(Beta, sum(z==k-1))
65  }
66  p = rdirichlet(1, Beta+1)
67  CHAIN_p = cbind(as.matrix(CHAIN_p),t(p))
68
69  cpost = sum(dnbinom(subset(x, z==k), rr, vv, log=T)) + log(vv*(1-vv)) +
      dgamma(rr, gammaprior[1], gammaprior[2], log=T)
70  rprop = rgamma(1, rr*3, 3)
71  vprop = rdirichlet(1, 30*c(vv, 1-vv))
72  npost = sum(dnbinom(subset(x, z==k), rprop, vprop[1], log=T)) + log(vprop[1]*
      (1-vprop[1])) + dgamma(rprop, gammaprior[1], gammaprior[2], log=T)
73  tempn = npost - dgamma(rprop, rr*3,3, log=T)- log(ddirichlet(vprop, 30*c(vv,
      1-vv)))
74  tempc = cpost - dgamma(rr, rprop*3,3, log=T)- log(ddirichlet(c(vv, 1-vv), 30*
      vprop))
75  if(log(runif(1)) <  tempn - tempc )
76  {
77    rr = rprop[1]
78    vv = vprop[1]
79  }
80  CHAIN_r = c(CHAIN_r, rr)
81  CHAIN_v = c(CHAIN_v, vv)
82
83  PP = matrix(0, n, Ncomponent)
84  PP[,1] = p[1] * dpois(x,lambda)
85  PP[,2] = p[2] * dnbinom(x, rr, vv)
86
87  PP = PP / rowSums(PP)
88  for(i in 1:n ){
89    z[i] = sample(1:Ncomponent, size=1, replace=T, PP[i,])-1
90              }
91    }
92  para.samples=as.matrix(cbind((CHAIN_lambda[(burnin+1):MCMCsteps]),(CHAIN_p
      [1,(burnin+1):MCMCsteps]),(CHAIN_r[(burnin+1):MCMCsteps]),(CHAIN_v[(burnin
      +1):MCMCsteps])))
93   colnames(para.samples) <- c("lambda","pie","r","v")
```

```
94      para=round(apply(para.samples, 2, mean), 4)
95      quan <- cbind(quantile(CHAIN_lambda[(burnin+1):MCMCsteps],prob=c
        (0.025,0.975)),quantile(CHAIN_p[1,(burnin+1):MCMCsteps],prob=c(0.025,0.975)
        ),quantile(CHAIN_r[(burnin+1):MCMCsteps],prob=c(0.025,0.975)),quantile(
        CHAIN_v[(burnin+1):MCMCsteps],prob=c(0.025,0.975)))
96      colnames(quan) <- c("lambda","pie","r","v")
97      return(list(parameters=para.samples,means=para,quantiles=quan))
98    }
99    scene1 <-POISNB(x,MCMCsteps=20000,burnin = 10000,prioralpha=2,priorbeta=4,
        gammaprior=c(15,1))
100   para <-scene1$means
101   lambda<-scene1$parameters[,1]
102   pie<-scene1$parameters[,2]
103   r<-scene1$parameters[,3]
104   v<-scene1$parameters[,4]
105   ############## plots ################################
106   plot(lambda,type="l",ylab="lambda",xlab="iterations",col="red")
107   plot(pie,type="l",ylab="pie",xlab="iterations",col="red")
108   plot(r,type="l",ylab="r",xlab="iterations",col="red")
109   plot(v,type="l",ylab="v",xlab="iterations",col="red")
```

C.1.2 Simulation studies in Scenario 2 (Section 4.3.2) and Scenario 3 (Section 4.3.3)

C.1.2.1 The proposed method

```
1  rm(list=ls())
2  library("gtools")
3  library("Rlab")
4  Newrbern = function(prob){
5      U = runif(length(prob))
6    return(1*(U<prob))
7      }
8  generate.data <- function(truep=c(0.6,0.1,0.1,0.1,0.1),truelambda=1,r=c
       (3,5,8,10),v=c(0.3,0.5,0.7,0.8),n=500)
9  {
10   x<-rep(0,n)
11   truez<-rep(0,n)
12   u<-runif(n)
13   for(i in 1:n){
14     if(u[i]<=0.6)
```

```
15      {
16      x[i]=rpois(1,truelambda)
17      truez[i]=0
18      }
19    else
20      if(0.6<u[i]&u[i]<=0.7){
21      x[i]=rnbinom(1,r[1],v[1])
22      truez[i]=1
23      }
24    else
25      if(0.7<u[i]&u[i]<=0.8){
26      x[i]=x[i]=rnbinom(1,r[2],v[2])
27      truez[i]=1
28      }
29    else
30      if(0.8<u[i]&u[i]<=0.9){
31      x[i]=rnbinom(1,r[3],v[3])
32      truez[i]=1
33      }
34    else{
35      x[i]=x[i]=rnbinom(1,r[4],v[4])
36      truez[i]=1
37      }
38    }
39    data<-list(s=truez, o=x)
40    return(data)
41  }
42  ndata <- generate.data(truep=c(0.6,0.1,0.1,0.1,0.1),truelambda=1,r=c(3,5,8,10),
        v=c(0.3,0.5,0.7,0.8),n=500)
43  x <-ndata$o
44  truez <- ndata$s
45
46  scenario2<-function(x,MCMCsteps=20000,burnin=10000,prioralpha=1,priorbeta=2)
47    {
48    p_start=c(0.8,0.2)
49    lambda_start=3
50    z_start=1-(x<=5)
51    n = length(x)
52    lambda=lambda_start
53    p=p_start
54    z=z_start
55    CHAIN_lambda=lambda
56    CHAIN_p=p[1]
57
```

```
58    for(step in 1:MCMCsteps)
59      {
60      print(c("iteration",step))
61      step
62
63      subxz1=subset(x,z==1)
64      freqmatrix = as.matrix(subxz1)%*%matrix(1,1,n)
65      freqx = rowSums(t(freqmatrix)==x)
66      Nprob = unique(cbind(x,freqx+1))
67      ft2 = rdirichlet(1,Nprob[,2])
68      f2 = colSums(t(as.matrix(ft2))%*%matrix(1,1,n) * (t(as.matrix(x)%*%matrix
        (1,1,length(ft2)))==Nprob[,1]))
69
70      a = p[1] * dpois(x,lambda)
71      b = p[2] * f2 + rep(0,n)
72      pq= a/(a+b)
73      z = 1 - Newrbern(pq)
74
75      GammaA=sum(x*(z==0))+prioralpha
76      GammaB = sum(z==0) + priorbeta
77      lambda = rgamma(1, GammaA+1, GammaB)
78      CHAIN_lambda = c(CHAIN_lambda,lambda)
79
80      BetaA = sum(z==0)
81      BetaB = sum(z==1)
82      p[1] = rbeta(1,BetaA+1, BetaB+1)
83      p[2] = 1-p[1]
84      CHAIN_p = c(CHAIN_p,p[1])
85      }
86    para.samples=as.matrix(cbind((CHAIN_lambda[(burnin+1):MCMCsteps]),(CHAIN_p[(
        burnin+1):MCMCsteps])))
87    colnames(para.samples) <- c("lambda","pie")
88    para=round(apply(para.samples, 2, mean), 4)
89    quan <- cbind(quantile(CHAIN_lambda[(burnin+1):MCMCsteps],prob=c(0.025,0.975)
        ),quantile(CHAIN_p[(burnin+1):MCMCsteps],prob=c(0.025,0.975)))
90    colnames(quan) <- c("lambda","pie")
91    classrate <- 1-(sum(diag(table(truez,z)))/sum(table(truez,z)))
92    return(list(parameters=para.samples,pq=pq,means=para,quantiles=quan,rate=
        classrate))
93  }
94    scene2 <-scenario2(x,MCMCsteps=20000,burnin=10000,prioralpha=1,priorbeta=2)
95    errorRate<-scene2$rate
96    pq <-scene2$pq
97    lambda<-scene2$parameters[,1]
```

```
  pie<-scene2$parameters[,2]
  quantiles<-scene2$quantiles
############## plots ################################
par(mfrow=c(1,2))
plot(lambda,type="l",ylab="lambda",xlab="iterations",col="red")
plot(pie,type="l",ylab="pie",xlab="iterations",col="red")
#### FDR ###################################
scene2FDR <- function(pq, thr=0.06){
fdr<-sum(subset(pq,pq<thr)/sum(pq<thr))
return(fdr=fdr)
}
FDR<-scene2FDR(pq, thr=0.06)
################ FNDR ######################
scene2FNDR <- function(pq, thr=0.06){
fndr<-sum(subset((1-pq),pq>=thr))/sum(pq>=thr)
return(fndr=fndr)
}
FNDR<-scene2FNDR(pq, thr=0.06)
```

C.1.2.2 Five-components mixture distribution

```
rm(list=ls())
library(gtools)
library(Rlab)
Newrbern = function(prob)
{
  U = runif(length(prob))
  return(1*(U<prob))
}
generate.data <- function(truep=c(0.6,0.1,0.1,0.1,0.1),truelambda=1,r=c
    (3,5,8,10),v=c(0.3,0.5,0.7,0.8),n=500)
  {
  x<-rep(0,n)
  truez<-rep(0,n)
  u<-runif(n)
  Ncomponent=5
  for(i in 1:n){
    if(u[i]<=0.6)
      {
    x[i]=rpois(1,truelambda)
    truez[i]=0
```

```
      }
  else
    if(0.6<u[i]&u[i]<=0.7){
      x[i]=rnbinom(1,r[1],v[1])
      truez[i]=1
      }
  else
    if(0.7<u[i]&u[i]<=0.8){
      x[i]=rnbinom(1,r[2],v[2])
      truez[i]=2
      }
  else
    if(0.8<u[i]&u[i]<=0.9){
      x[i]=rnbinom(1,r[3],v[3])
      truez[i]=3
      }
  else{
    x[i]=rnbinom(1,r[4],v[4])
    truez[i]=4
     }
  }
  data<-list(s=truez, o=x)
  return(data)
}
ndata <- generate.data(truep=c(0.6,0.1,0.1,0.1,0.1),truelambda=1,r=c(3,5,8,10),
    v=c(0.3,0.5,0.7,0.8),n=500)
x <-ndata$o
truez <- ndata$s

scenario2 <- function(x,MCMCsteps=20000,burnin=10000,prioralpha=2,priorbeta=1,
    gammaprior=c(20,1))
{
  p_start=c(0.6,0.1,0.1,0.1,0.1)
  lambda_start=3
  r_start=c(8,10, 11, 12)
  v_start=c(0.2,0.4, 0.3, 0.5)
  z_start=1-(x<=5)
  n = length(x)
  Ncomponent = 5
  lambda=lambda_start
  p=p_start
  z=z_start
  rr = r_start
  vv = v_start
```

```
CHAIN_lambda=lambda_start
CHAIN_p=p_start
CHAIN_r=r_start
CHAIN_v=v_start
z=z_start

for(step in 1:MCMCsteps)
{
  print(c("iteration",step))
  step

  GammaA=sum(x*(z==0))+prioralpha
  GammaB = sum(z==0) + priorbeta
  lambda = rgamma(1, GammaA+1, GammaB)
  CHAIN_lambda = c(CHAIN_lambda,lambda)

  Beta=NULL
  for(k in 1:Ncomponent){
    Beta = c(Beta, sum(z==k-1))
  }
  p = rdirichlet(1, Beta+1)

  CHAIN_p = cbind(as.matrix(CHAIN_p),t(p))

  for(k in 1: (Ncomponent-1)){
    cpost = sum(dnbinom(subset(x, z==k), rr[k], vv[k], log=T)) + log(vv[k]*
  (1-vv[k])) + dgamma(rr[k], gammaprior[1], gammaprior[2], log=T)
    rprop = rgamma(1, rr[k]*3, 3)
    vprop = rdirichlet(1, 50*c(vv[k], 1-vv[k]))
    npost = sum(dnbinom(subset(x, z==k), rprop, vprop[1], log=T)) + log(vprop
  [1]*(1-vprop[1])) + dgamma(rprop, gammaprior[1], gammaprior[2], log=T)
    tempn = npost - dgamma(rprop, rr[k]*3,3, log=T)-log(ddirichlet(vprop, 50*
  c(vv[k], 1-vv[k])))
    tempc = cpost - dgamma(rr[k], rprop*3,3, log=T)-log(ddirichlet(c(vv[k],
  1-vv[k]), 50*vprop))
    if(log(runif(1)) <  tempn - tempc )
    {
      rr[k] = rprop[1]
      vv[k] = vprop[1]
    }
  }
  CHAIN_r = cbind(CHAIN_r, rr)
  CHAIN_v = cbind(CHAIN_v, vv)

```

```
102       PP = matrix(0, n, Ncomponent)
103       PP[,1] = p[1] * dpois(x,lambda)
104       for(k in 2:Ncomponent)
105       {
106         PP[,k] = p[k] * dnbinom(x, rr[k-1], vv[k-1])
107       }
108       PP = PP / rowSums(PP)
109       for(i in 1:n ){
110         z[i] = sample(1:Ncomponent, size=1, replace=T, PP[i,])-1
111       }
112     }
113     para.samples=as.matrix(cbind((CHAIN_lambda[(burnin+1):MCMCsteps]),(CHAIN_p
          [1,(burnin+1):MCMCsteps])))
114     colnames(para.samples) <- c("lambda","pie")
115     para=round(apply(para.samples, 2, mean), 4)
116     quan <- cbind(quantile(CHAIN_lambda[(burnin+1):MCMCsteps],prob=c(0.025,0.975)
          ),quantile(CHAIN_p[1,(burnin+1):MCMCsteps],prob=c(0.025,0.975)))
117     colnames(quan) <- c("lambda","pie")
118     classrate <- ((sum(table(z,truez)[1,])+sum(table(z,truez)[,1]) - 2*table(z,
          truez)[1,1])/(sum(table(z,truez))))
119     return(list(parameters=para.samples,PP=PP,means=para,quantiles=quan,rate=
          classrate))
120   }
121   scene2 <-scenario2(x,MCMCsteps=20000,burnin = 10000,prioralpha=2,priorbeta=1)
122   para <-scene2$means
123   errorRate<-scene2$rate
124   PP <-scene2$PP
125   lambda<-scene2$parameters[,1]
126   pie<-scene2$parameters[,2]
127   quantiles<-scene2$quantiles
128   ######## plots ########################
129   plot(lambda,type="l",ylab="lambda",xlab="iterations",col="red")
130   plot(pie,type="l",ylab="pie",xlab="iterations",col="red")
131   #### FDR ###################################
132   scene2FDR <- function(PP, thr=0.08){
133     fdr<-sum(subset(PP[,1],PP[,1]<thr)/sum(PP[,1]<thr))
134     return(fdr=fdr)
135   }
136   FDR<-scene2FDR(PP, thr=0.08)
137   ################ FNDR ######################
138   scene2FNDR <- function(PP, thr=0.08){
139     fndr<-sum(subset((1-PP[,1]),PP[,1]>=thr))/sum(PP[,1]>=thr)
140     return(fndr=fndr)
141   }
```

```
FNDR<-scene2FNDR(PP, thr=0.08)
```

C.1.3 Data analysis for mixture model

```
rm(list=ls())
library(enRich)
data(p300cbp.1000bp)
genetic_data<-p300cbp.1000bp
write.csv(genetic_data,"genetic data.csv")

Chrome21<-read.table("genetic data.csv", sep=',', header=TRUE)
str(Chrome21)
summary(Chrome21)
x <- Chrome21$count.CBPT301
n = length(x)

scenario1<-function(x,MCMCsteps=20000,burnin=10000,prioralpha=1,priorbeta=2)
{
    n=length(x)
    p_start=c(0.8,0.2)
    lambda_start= 10
    z_start=1-(x<=8)
    lambda=lambda_start
    p=p_start
    z=z_start
    CHAIN_lambda=lambda
    CHAIN_p=p[1]

  Newrbern = function(prob)
      {
    U = runif(length(prob))
    return(1*(U<prob))
      }
  for(step in 1:MCMCsteps)
    {
    print(c("iteration",step))
    step
    subxz1=subset(x,z==1)
    freqmatrix = as.matrix(subxz1)%*%matrix(1,1,n)

    freqx = rowSums(t(freqmatrix)==x)
```

```
38      Nprob = unique(cbind(x,freqx+1))
39      ft2 = rdirichlet(1,Nprob[,2])
40      f2 = colSums(t(as.matrix(ft2))%*%matrix(1,1,n)*(t(as.matrix(x)
41      %*%matrix(1,1,length(ft2)))==Nprob[,1]))
42
43      a = p[1] * dpois(x,lambda)
44      b = p[2] * f2 + rep(0,n)
45      pq= a/(a+b)
46      z = 1 - Newrbern(pq)
47
48      GammaA=sum(x*(z==0))+prioralpha
49      GammaB = sum(z==0) + priorbeta
50      lambda = rgamma(1, GammaA+1, GammaB)
51      CHAIN_lambda = c(CHAIN_lambda,lambda)
52
53      BetaA = sum(z==0)
54      BetaB = sum(z==1)
55      p[1] = rbeta(1,BetaA+1, BetaB+1)
56      p[2] = 1-p[1]
57      CHAIN_p = c(CHAIN_p,p[1])
58      }
59    pq=as.matrix(cbind(pq))
60    return(pq=pq)
61    }
62    scene1 <-scenario1(x,MCMCsteps=20000,burnin=10000,prioralpha=1,priorbeta=2)
63    pq<-scene1
64 ################  Enrich regions  ##################
65    genome_wide <- function(pq, thr=0.01){
66    fdr<-sum(subset(pq,pq<thr)/sum(pq<thr))
67    enrich<-which(pq<thr)
68    sum.enrich <-sum(pq<thr)
69    return(list(fdr=fdr,enrich=enrich,sum.enrich=sum.enrich))
70    }
71    genome.mixmod<-genome_wide(pq, thr=0.01)
72    fdr<-genome.mixmod$fdr
73    enrich<-genome.mixmod$enrich
74    sum.enrich<-genome.mixmod$sum.enrich
```

C.2 R code for Markov random field model

C.2.1 Simulation studies in Scenario 1: NB distribution for the noise component

```
rm(list=ls())
library("gtools")
library("gamlss.dist")
generate.data <- function(truepii = c(0.5,0.5),truep1=matrix(c(0.8,0.2,0.3,0.7)
    ,byrow=TRUE,nrow=2),v=c(0.6,0.1),r=c(5,12),n = 500)
{
  truez<-rep(0, n)
  x<-rep(0, n)
  truez[1]<-rbinom(1, 1, truepii[1])
  for (i in 2:n)
    {
    if (truez[i-1]==0)
    truez[i]<-rbinom(1, 1,  truep1[1, 2])
    else
    truez[i]<-rbinom(1, 1, truep1[2, 2])
    }
  for (i in 1:n)
   {
    if (truez[i]==0)
    {
    x[i]<-rnbinom(1, r[1], v[1])
    }
  else
    {
    x[i]<-rnbinom(1, r[2], v[2])
    }
   }
  data<-list(s=truez, o=x)
  return(data)
}
  ndata <- generate.data(truepii = c(0.5,0.5),truep1=matrix(c(0.8,0.2,0.3,0.7),
    byrow=TRUE,nrow=2),v=c(0.6,0.1),r=c(5,12),n = 500)
  x<-ndata$o
  truez<-ndata$s

mrf_NB<- function(x,MCMCsteps=20000,burnin=10000, Pprior=c( 1, 1, 0.5, 0.5),
    gammaprior=c(5,1))
{
```

```
36
37  Ncomponent = 2
38  p1_start<- matrix(c(0.7,0.3,0.4,0.6),byrow=TRUE,nrow=2)
39  r_start = 3
40  v_start = 0.2
41  P1_start = 0.2
42  P2_start = 0.4
43  z_start<- 1-(x<=5)
44  n=length(x)
45  p=p1_start
46  rr = r_start
47  vv = v_start
48  P1 = P1_start
49  P2 = P2_start
50  CHAIN_P1 = P1_start
51  CHAIN_P2 = P2_start
52  CHAIN_r=r_start
53  CHAIN_v=v_start
54  z=z_start
55
56  for(step in 1:MCMCsteps)
57  {
58    print(c("iteration",step))
59    step
60
61    cpost = sum(dnbinom(subset(x, z==0), rr, vv, log=T)) + log(vv*(1-vv)) +
    dgamma(rr, gammaprior[1], gammaprior[2], log=T)
62    rprop = rgamma(1, rr*2, 2)
63    vprop = rdirichlet(1, 30*c(vv, 1-vv))
64    npost = sum(dnbinom(subset(x, z==0), rprop, vprop[1], log=T)) + log(vprop
    [1]*(1-vprop[1])) + dgamma(rprop,gammaprior[1], gammaprior[2], log=T)
65    tempn = npost - dgamma(rprop, rr*2,2, log=T)- log(ddirichlet(vprop, 30*c(vv
    , 1-vv)))
66    tempc = cpost - dgamma(rr, rprop*2,2, log=T)- log(ddirichlet(c(vv, 1-vv),
    30*vprop))
67    if(log(runif(1)) <  tempn - tempc )
68    {
69    rr = rprop[1]
70    vv = vprop[1]
71    }
72    CHAIN_r = c(CHAIN_r, rr)
73    CHAIN_v = c(CHAIN_v, vv)
74
75    subxz1=subset(x,z==1)
```

```
76      freqmatrix = as.matrix(subxz1)%*%matrix(1,1,n)
77      freqx = rowSums(t(freqmatrix)==x)
78      Nprob = unique(cbind(x,freqx+1))
79      ft2 = rdirichlet(1,Nprob[,2])
80      f2 = colSums(t(as.matrix(ft2))%*%matrix(1,1,n) * (t(as.matrix(x)%*%matrix
        (1,1,length(ft2)))==Nprob[,1]))
81
82      estPi <- table(z[-length(z)], z[-7])
83      rowtotal <- estPi %*% matrix(1, nrow=nrow(p), ncol=1)
84      Pi <- diag(as.vector(1/rowtotal)) %*% estPi
85
86      BetaA=estPi[1,2]+Pprior[1]
87      BetaB = estPi[1,1]+ Pprior[2]
88      P1 = rbeta(1, BetaA+1, BetaB+1)
89      CHAIN_P1 = c(CHAIN_P1,P1)
90
91      BetaC = estPi[2,2]+Pprior[3]
92      BetaD = estPi[2,1]+Pprior[4]
93      P2 = rbeta(1,BetaC+1,BetaD+1)
94      CHAIN_P2 = c(CHAIN_P2,P2)
95
96      PP = matrix(0, n, Ncomponent)
97      for (i in 1:n){
98              if(i==1)
99             {
100               PP[1,1] = (Pi[1,2])/(Pi[1,2]+Pi[2,1])
101               PP[1,2] = (Pi[2,1])/(Pi[1,2]+Pi[2,1])
102            }
103             else{
104         PP[i,1] = Pi[1,2]*Pi[1,1] * dnbinom(x[i], rr, vv)
105         PP[i,2] = Pi[2,2]*Pi[2,1]* f2[i]
106         }
107       }
108     PP = PP / rowSums(PP)
109       for(i in 1:n ){
110       z[i] = sample(1:Ncomponent, size=1, replace=T, PP[i,])-1
111       }
112     }
113     para.samples <-as.matrix(cbind((CHAIN_P1[(burnin+1):MCMCsteps]),(CHAIN_P2[(
        burnin+1):MCMCsteps]),(CHAIN_v[(burnin+1):MCMCsteps]),(CHAIN_r[(burnin+1):
        MCMCsteps])))
114     colnames(para.samples) <- c("P1","P2","v","r")
115     para <- round(apply(para.samples,2,mean),4)
```

```
116     quan <- cbind(quantile(CHAIN_P1[(burnin+1):MCMCsteps],prob=c(0.025,0.975)),
        quantile(CHAIN_P2[(burnin+1):MCMCsteps],prob=c(0.025,0.975)),quantile(
        CHAIN_v[(burnin+1):MCMCsteps],prob=c(0.025,0.975)),quantile(CHAIN_r[(burnin
        +1):MCMCsteps],prob=c(0.025,0.975)))
117     colnames(quan) <- c("P1","P2","v","r")
118     classrate <- 1-(sum(diag(table(truez,z))))/(sum(table(truez,z)))
119     return(list(parameters=para.samples,mean=para,quantiles=quan,errorRate=
        classrate))
120 }
121   NB_mrf <- mrf_NB(x,MCMCsteps=20000,burnin=10000, Pprior=c( 1, 1, 0.5,0.5),
      gammaprior=c(5,2))
122   NB_mrf
```

C.2.2 Simulation studies in Scenario 2: ZIP distribution for the noise component

```
1
2 rm(list=ls())
3 library("gtools")
4 library("gamlss.dist")
5
6 generate.data<-function(truepii = c(0.1,0.9),truep1 = matrix(c(0.7,0.3,0.2,0.8)
      ,byrow=TRUE,nrow=2),truelambda=2,truepie = 0.5,v =0.6,r=75,n = 500)
7   {
8   truez<-rep(0, n)
9   x<-rep(0, n)
10  truez[1]<-rbinom(1, 1, truepii[1])
11  for (i in 2:n)
12  {
13      if (truez[i-1]==0)
14  truez[i]<-rbinom(1, 1,  truep1[1, 2])
15  else
16  truez[i]<-rbinom(1, 1, truep1[2, 2])
17    }
18     for (i in 1:n)
19     {
20     if (truez[i]==0)
21     {
22     x[i]<-rZIP(1, truelambda, truepie)
23
24     }
```

```
25      else
26      {
27      x[i]<-rnbinom(1, r,v)
28      }
29      }
30    data<-list(s=truez, o=x)
31    return(data)
32   }
33    ndata <- generate.data(truepii = c(0.1,0.9),truep1 = matrix(c
        (0.7,0.3,0.2,0.8),byrow=TRUE,nrow=2),truelambda=2,truepie = 0.5,v =0.6,r
        =75,n = 500)
34    x <-ndata$o
35    truez <- ndata$s
36
37  mrf_ZIP <- function(x,MCMCsteps=20000,burnin = 10000,Pprior=c( 1, 1, 0.5,0.5),
      gammaprior= c(4,2),pieprior=c(4,2))
38  {
39      p1_start<- matrix(c(0.8,0.2,0.5,0.5),byrow=TRUE,nrow=2)
40      pie_start = 0.2
41      lambda_start= 3
42      Ncomponent = 2
43      P1_start = 0.1
44      P2_start = 0.2
45      y_start <- 1-(x<=5)
46      z_start<- 1-(x<=5)
47      n = length(x)
48      p=p1_start
49      lambda=lambda_start
50      pie = pie_start
51      P1 = P1_start
52      P2 = P2_start
53      CHAIN_lambda=lambda
54      CHAIN_P1 = P1
55      CHAIN_P2 = P2
56      z=z_start
57      y=y_start
58      CHAIN_pie=pie
59
60    for(step in 1:MCMCsteps)
61      {
62        print(c("iteration",step))
63        step
64
65      estPi <- table(z[-length(z)], z[-1])
```

```
    rowtotal <- estPi %*% matrix(1, nrow=nrow(p), ncol=1)
    Pi <- diag(as.vector(1/rowtotal)) %*% estPi

    BetaA = estPi[1,2]+Pprior[1]
    BetaB = estPi[1,1]+ Pprior[2]
    P1 = rbeta(1, BetaA+1, BetaB+1)
    CHAIN_P1 = c(CHAIN_P1,P1)

    BetaC = estPi[2,2]+Pprior[3]
    BetaD = estPi[2,1]+Pprior[4]
    P2 = rbeta(1,BetaC+1,BetaD+1)
    CHAIN_P2 = c(CHAIN_P2,P2)

    subxz1=subset(x,z==1)
    freqmatrix = as.matrix(subxz1)%*%matrix(1,1,n)
    freqx = rowSums(t(freqmatrix)==x)
    Nprob = unique(cbind(x,freqx+1))
    ft2 = rdirichlet(1,Nprob[,2])
    f2 = colSums(t(as.matrix(ft2))%*%matrix(1,1,n) * (t(as.matrix(x)%*%matrix
   (1,1,length(ft2)))==Nprob[,1]))

    pieA = sum(z==0&y==0)+pieprior[1]
    pieB = sum(z==0&y==1)+pieprior[2]
    pie = rbeta(1,pieB+1,pieA+1)
    CHAIN_pie = c(CHAIN_pie,pie)

    a <- ifelse(x==0,1,0)*pie
    b <- dpois(x,lambda)*(1-pie)
    pq = b/(a+b)
    y  <-rbinom(n,1,pq)

  PP = matrix(0, n, Ncomponent)
  for (c in 1:n){
   if(c==1)
        {
        PP[1,1] = (Pi[1,2])/(Pi[1,2]+Pi[2,1])
        PP[1,2] = (Pi[2,1])/(Pi[1,2]+Pi[2,1])
         }
     else{
    PP[c,1] = P1*(1-P1)* dZIP(x[c], lambda, pie)
    PP[c,2] = P2*(1-P2)* f2[c]
    }
    }
  PP = PP / rowSums(PP)
```

```
    for(i in 1:n ){
    z[i] = sample(1:Ncomponent, size=1, replace=T, PP[i,])-1
     }

   GammaA = sum(x*(z==0&y==1))+gammaprior[1]
   GammaB = sum(z==0&y==1)+gammaprior[2]
   lambda = rgamma(1, GammaA+1, GammaB)
      CHAIN_lambda = c(CHAIN_lambda,lambda)
    }
   para.samples=as.matrix(cbind((CHAIN_P1[(burnin+1):MCMCsteps]),(CHAIN_P2[(
     burnin+1):MCMCsteps]),(CHAIN_lambda[(burnin+1):MCMCsteps]),(CHAIN_pie[(
     burnin+1):MCMCsteps])))
   colnames(para.samples) <- c("P1","P2","lambda","pie")
   para=round(apply(para.samples, 2, mean), 4)
   quan <- cbind(quantile(CHAIN_P1[(burnin+1):MCMCsteps],prob=c(0.025,0.975)),
     quantile(CHAIN_P2[(burnin+1):MCMCsteps],prob=c(0.025,0.975)),quantile(
     CHAIN_lambda[(burnin+1):MCMCsteps],prob=c(0.025,0.975)),quantile(CHAIN_pie
     [(burnin+1):MCMCsteps],prob=c(0.025,0.975)))
   colnames(quan) <- c("P1","P2","lambda","pie")
   classrate <- 1-(sum(diag(table(truez,z))))/(sum(table(truez,z)))
   return(list(parameters=para.samples,means=para,quantiles=quan,rate=classrate)
     )
 }
ZIP_mrf <- mrf_ZIP(x,MCMCsteps=20000,burnin = 10000,Pprior=c( 1, 1, 0.5,0.5),
    gammaprior= c(4,2),pieprior=c(4,2))
ZIP_mrf

############## plots ################
 plot(CHAIN_P1,type="l",ylab="P1",xlab="iterations",col="red")
 plot(CHAIN_P2,type="l",ylab="P2",xlab="iterations",col="red")
 plot(V,type="l",ylab="v",xlab="iterations",col="red")
 plot(CHAIN_r,type="l",ylab="r",xlab="iterations",col="red")
 plot(CHAIN_pie,type="l",ylab="pie",xlab="iterations",col="red")
```

C.2.3 Simulation studies in Scenario 3: ZINB distribution for the noise component

```

rm(list=ls())
library("gtools")
library("gamlss.dist")
```

```
generate.data <- function(truepii = c(0.1,0.9),truep1=matrix(c(0.7,0.3,0.2,0.8)
    ,byrow=TRUE,nrow=2),truepie=0.5,v=c(0.2,0.6),r=c(3,37),n =500)
{
  truez<-rep(0, n)
  x<-rep(0, n)
  truez[1]<-rbinom(1, 1, truepii[1])
    for (i in 2:n)
    {
     if (truez[i-1]==0)
     truez[i]<-rbinom(1, 1,  truep1[1, 2])
     else
     truez[i]<-rbinom(1, 1, truep1[2, 2])
    }
     for (i in 1:n)
  {
     if (truez[i]==0)
     {
     x[i]<-rZINBI(1, r[1],v[1], truepie)
     }
  else
  {
     x[i]<-rnbinom(1, r[2], v[2])

  }
  }
data<-list(s=truez, o=x)
return(data)
}
  ndata <- generate.data(truepii = c(0.1,0.9),truep1=matrix(c(0.7,0.3,0.2,0.8),
    byrow=TRUE,nrow=2),truepie=0.5,v=c(0.2,0.6),r=c(3,37),n =500)
  x <- ndata$o
  truez <-ndata$s

mrf_ZINB <- function(x,MCMCsteps=20000,burnin = 10000,Pprior=c( 1, 1, 0.5, 0.5)
    ,pieprior=c(1,2), gammaprior=c(7,1))
{
     Ncomponent = 2
     p1_start<- matrix(c(0.7,0.3,0.5,0.5),byrow=TRUE,nrow=2)
     pie_start = 0.2
     r_start = 3
     v_start = 0.2
     P1_start = 0.2
     P2_start = 0.4
     y_start <- 1-(x<=7)
```

```
46      z_start<- 1-(x<=7)
47      n= length(x)
48      p=p1_start
49      pie = pie_start
50      rr = r_start
51      vv = v_start
52      P1 = P1_start
53      P2 = P2_start
54      CHAIN_P1 = P1
55      CHAIN_P2 = P2
56      CHAIN_r=rr
57      CHAIN_v=vv
58      z=z_start
59      y=y_start
60      CHAIN_pie=pie
61
62      for(step in 1:MCMCsteps)
63      {
64        print(c("iteration",step))
65        step
66
67        cpost = sum(dnbinom(subset(x,z==0&y==1), rr, vv, log=T)) + log(vv*(1-vv))
     + dgamma(rr, gammaprior[1], gammaprior[2], log=T)
68        rprop = rgamma(1, rr*2,2)
69        vprop = rdirichlet(1, 30*c(vv, 1-vv))
70        npost = sum(dnbinom(subset(x,z==0&y==1), rprop, vprop[1], log=T)) + log(
    vprop[1]*(1-vprop[1])) + dgamma(rprop, gammaprior[1], gammaprior[2], log=T)
71        tempn = npost - dgamma(rprop, rr*2,2, log=T)- log(ddirichlet(vprop, 30*c(
    vv, 1-vv)))
72        tempc = cpost - dgamma(rr, rprop*2,2, log=T)- log(ddirichlet(c(vv, 1-vv),
     30*vprop))
73        if(log(runif(1)) <  tempn - tempc )
74        {
75        rr = rprop[1]
76        vv = vprop[1]
77        }
78        CHAIN_r = c(CHAIN_r, rr)
79        CHAIN_v = c(CHAIN_v, vv)
80
81        subxz1=subset(x,z==1)
82        freqmatrix = as.matrix(subxz1)%*%matrix(1,1,n)
83        freqx = rowSums(t(freqmatrix)==x)
84        Nprob = unique(cbind(x,freqx+1))
85        ft2 = rdirichlet(1,Nprob[,2])
```

```
86        f2 = colSums(t(as.matrix(ft2))%*%matrix(1,1,n) * (t(as.matrix(x)%*%matrix
      (1,1,length(ft2)))==Nprob[,1]))
87
88      estPi <- table(z[-length(z)], z[-1])
89      rowtotal <- estPi %*% matrix(1, nrow=nrow(p), ncol=1)
90      Pi <- diag(as.vector(1/rowtotal)) %*% estPi
91
92      BetaA=estPi[1,2]+Pprior[1]
93      BetaB = estPi[1,1]+ Pprior[2]
94      P1 = rbeta(1, BetaA+1, BetaB+1)
95      CHAIN_P1 = c(CHAIN_P1,P1)
96
97      BetaC = estPi[2,2]+Pprior[3]
98      BetaD = estPi[2,1]+Pprior[4]
99      P2 = rbeta(1,BetaC+1,BetaD+1)
100     CHAIN_P2 = c(CHAIN_P2,P2)
101
102     pieA = sum(z==0&y==0)+pieprior[1]
103     pieB = sum(z==0&y==1)+pieprior[2]
104     pie = rbeta(1,pieB+1,pieA+1)
105     CHAIN_pie = c(CHAIN_pie,pie)
106
107     a <- ifelse(x==0,1,0)*pie
108     b <- (1-pie)*dnbinom(x,rr,vv)
109     pq = b/(a+b)
110     y = rbinom(n,1,pq)
111
112   PP = matrix(0, n, Ncomponent)
113   for (c in 1:n){
114     if(c==1)
115           {
116         PP[1,1] = (Pi[1,2])/(Pi[1,2]+Pi[2,1])
117         PP[1,2] = (Pi[2,1])/(Pi[1,2]+Pi[2,1])
118          }
119         else{
120   PP[c,1] = Pi[1,2]*Pi[1,1]* dZINBI(x[c], rr,vv, pie, log = FALSE)
121   PP[c,2] = Pi[2,2]*Pi[2,1]* f2[c]
122       }
123      }
124     PP = PP / rowSums(PP)
125     for(i in 1:n ){
126     z[i] = sample(1:Ncomponent, size=1, replace=T, PP[i,])-1
127         }
128   }
```

```
129   para.samples <-as.matrix(cbind((CHAIN_P1[(burnin+1):MCMCsteps]),(CHAIN_P2[(
        burnin+1):MCMCsteps]),(CHAIN_v[(burnin+1):MCMCsteps]),(CHAIN_r[(burnin+1):
        MCMCsteps]),(CHAIN_pie[(burnin+1):MCMCsteps])))
130   colnames(para.samples) <- c("P1","P2","v","r","pie")
131   para <- round(apply(para.samples,2,mean),4)
132   quan <- cbind(quantile(CHAIN_P1[(burnin+1):MCMCsteps],prob=c(0.025,0.975)),
        quantile(CHAIN_P2[(burnin+1):MCMCsteps],prob=c(0.025,0.975)),quantile(
        CHAIN_v[(burnin+1):MCMCsteps],prob=c(0.025,0.975)),quantile(CHAIN_r[(burnin
        +1):MCMCsteps],prob=c(0.025,0.975)),quantile(CHAIN_pie[(burnin+1):MCMCsteps
        ],prob=c(0.025,0.975)))
133   colnames(quan) <- c("P1","P2","v","r","pie")
134   classrate <- 1-(sum(diag(table(truez,z))))/(sum(table(truez,z)))
135   return(list(parameters=para.samples,mean=para,quantiles=quan,rate=classrate))
136
137  }
138 ZINB_mrf <- mrf_ZINB(x,MCMCsteps=20000,burnin = 10000,Pprior=c( 1, 1, 0.5, 0.5)
      ,pieprior=c(1,2), gammaprior=c(7,1))
139 ZINB_mrf
```

C.2.4 Data analysis for MRF model

```
1  rm(list=ls())
2  library("parallel")
3  library("enRich")
4  data(p300cbp.200bp)
5  genetic_data<-p300cbp.200bp
6  write.csv(genetic_data,"genetic data.csv")
7
8  Chrome21<-read.csv("genetic data.csv", sep=',', header=TRUE)
9  str(Chrome21)
10 summary(Chrome21)
11 x <- Chrome21$count.p300T301[1:234720]
12 n = length(x)
13
14 mrf_ZIP <- function(x,MCMCsteps=20000,burnin = 10000,Pprior=c( 2,2,1,1),
       pieprior=c(2,1), gammaprior=c(4,2))
15 {
16
17   for(step in 1:MCMCsteps)
18   {
19     p1_start<- matrix(c(0.8,0.2,0.5,0.5),byrow=TRUE,nrow=2)
```

```
20      pie_start = 0.002
21      lambda_start= 4
22      Ncomponent = 2
23      P1_start = 0.001
24      P2_start = 0.04
25      y_start <- 1-(x<=7)
26      z_start<- 1-(x<=7)
27      n = length(x)
28      p=p1_start
29      lambda=lambda_start
30      pie = pie_start
31      P1 = P1_start
32      P2 = P2_start
33      CHAIN_lambda=lambda
34      CHAIN_P1 = P1
35      CHAIN_P2 = P2
36      z=z_start
37      y=y_start
38      CHAIN_pie=pie
39
40      estPi <- table(z[-length(z)], z[-1])
41      rowtotal <- estPi %*% matrix(1, nrow=nrow(p), ncol=1)
42      Pi <- diag(as.vector(1/rowtotal)) %*% estPi
43
44      BetaA = estPi[1,2]+Pprior[1]
45      BetaB = estPi[1,1]+ Pprior[2]
46      P1 = rbeta(1, BetaA+1, BetaB+1)
47      CHAIN_P1 = c(CHAIN_P1,P1)
48
49      BetaC = estPi[2,2]+Pprior[3]
50      BetaD = estPi[2,1]+Pprior[4]
51      P2 = rbeta(1,BetaC+1,BetaD+1)
52      CHAIN_P2 = c(CHAIN_P2,P2)
53
54      subxz1=subset(x,z==1)
55      freqmatrix = as.matrix(subxz1)%*%matrix(1,1,n)
56      freqx = rowSums(t(freqmatrix)==x)
57      Nprob = unique(cbind(x,freqx+1))
58      ft2 = rdirichlet(1,Nprob[,2])
59      f2 = colSums(t(as.matrix(ft2))%*%matrix(1,1,n) * (t(as.matrix(x)%*%matrix
        (1,1,length(ft2)))==Nprob[,1]))
60
61      pieA = sum(z==0&y==0)+pieprior[1]
62      pieB = sum(z==0&y==1)+pieprior[2]
```

```
63      pie = rbeta(1,pieB+1,pieA+1)
64      CHAIN_pie = c(CHAIN_pie,pie)
65
66      a <- ifelse(x==0,1,0)*pie
67      b <- dpois(x,lambda)*(1-pie)
68      pq = b/(a+b)
69      y  <-rbinom(n,1,pq)
70
71      GammaA = sum(x*(z==0&y==1))+gammaprior[1]
72      GammaB = sum(z==0&y==1)+gammaprior[2]
73      lambda = rgamma(1, GammaA+1, GammaB)
74      CHAIN_lambda = c(CHAIN_lambda,lambda)
75
76      PP = matrix(0, n, Ncomponent)
77      for (c in 1:n){
78        if(c==1)
79                  {
80               PP[1,1] = (Pi[1,2])/(Pi[1,2]+Pi[2,1])
81               PP[1,2] = (Pi[2,1])/(Pi[1,2]+Pi[2,1])
82                }
83               else{
84          PP[c,1] = P1*(1-P1)* dZIP(x[c], lambda, pie)
85          PP[c,2] = P2*(1-P2)* f2[c]
86          }
87      }
88      PP = PP / rowSums(PP)
89      for(i in 1:n ){
90        z[i] = sample(1:Ncomponent, size=1, replace=T, PP[i,])-1
91      }
92    }
93    return(PP=PP)
94  }
95
96  subx <- split(x, ceiling(seq_along(x)/78240))
97  mynodes <-  makeCluster(20)
98  clusterEvalQ(mynodes,library(gtools))
99  clusterEvalQ(mynodes,library(gamlss.dist))
100 parx <- parLapply(mynodes,subx,mrf_ZIP)
101 stopCluster(mynodes)
102
103 col1 <- lapply(parx,function(x) x[,1])
104 col2 <- lapply(parx,function(x) x[,2])
105 PPs <- matrix(cbind(unlist(col1),unlist(col2)),ncol=2)
106
```

```
genome_wide <- function(PPs, thr=0.005){
   fdr<-sum(subset(PPs[,1],PPs[,1]<thr)/sum(PPs[,1]<thr))
   enrich<-which(PPs[,1]<thr)
   sum.enrich <-sum(PPs[,1]<thr)
   return(list(fdr=fdr,enrich=enrich,sum.enrich=sum.enrich))
}
genome.mrf<-genome_wide(PPs, thr=0.005)
fdr<-genome.mrf$fdr
enrich<-genome.mrf$enrich
sum.enrich<-genome.mrf$sum.enrich
```