Finite Bayesian Mixture Models with Applications in Spatial Cluster Analysis and Bioinformatics

Dissertation

by

Martin Schäfer

Submitted to
the Faculty of Statistics
of the TU Dortmund University

in Partial Fulfillment of
the Requirements for the Degree of
Doktor der Naturwissenschaften

Dortmund, November 2015

Referees:

Prof. Dr. Katja Ickstadt

Prof. Dr. Jörg Rahnenführer

Date of Oral Examination: November 6, 2015

Abstract

In many statistical applications, one encounters populations that form homogenous subgroups regarding one or several characteristics. Across the subgroups, however, heterogeneity may often be found. Mixture distributions are a natural means to model data from such applications.

This PhD thesis is based on two projects that focus on such applications. In the first project, spatial nanoscale clusters formed by Ras proteins in the cell membrane are investigated. Such clusters play a crucial role in intracellular communication and are thus of interest in cancer research. In this case, the subgroups are clustered and non-clustered proteins.

In the second project, epigenomic data obtained from sequencing experiments are integrated with another genomic or epigenomic input, aiming, e.g., to detect genes that contribute to the development of cancer. Here, the subgroups are defined by a) genes presenting congruent (epi)genomic aberrations in both considered variables, b) genes presenting incongruent aberrations, and c) genes lacking aberrations in at least one of the variables.

Employing a Bayesian framework, objects are classified in both projects by fitting finite univariate mixture distributions with a small fixed number of components to values from a score summarizing relevant information about the research question. Such mixture distributions have favorable characteristics in terms of interpretation and present little sensitivity to label switching in Markov Chain Monte Carlo analyses. Mixtures of gamma distributions are considered for Ras proteins, while mixtures of normal and exponential or gamma distributions are a focus for the bioinformatic analysis. In the latter, classification is the primary goal, while in the Ras protein application, estimating key parameters of the spatial clustering is of more interest.

The results of both projects are presented in this thesis. For both applications, the methods have been implemented in software and their performance is compared with competing approaches on experimental as well as on simulated

data. To warrant an appropriate simulation of Ras protein patterns, a new cluster point process model called the double Matérn cluster process is developed and described in this thesis.

Acknowledgements

First of all, I wish to thank my advisor Katja Ickstadt for her guidance during my PhD time, in particular for giving me both freedom in many research problems and valuable advice whenever needed, but also for providing an always very comfortable working atmosphere in her research group.

Alan E. Gelfand supported my research on spatial Ras protein clusters through helpful discussions on concepts. I therefore would like to thank him for advice and for sharing his large statistical experience.

I would like to thank Peter Verveer and members of his former research group at the Max Planck Institute of Molecular Physiology in Dortmund for providing the problem regarding the research on Ras protein clusters, as well as for the fruitful collaboration in general. In particular, I want to thank Yvonne Radon for furnishing me with experimental Ras data and Thomas Klein for making my MATLAB code more efficient.

My research on spatial Ras protein clusters formed part of the project 'Modeling spatial effects of cellular signals' in the Research Training Group 1032/2 'Statistical Modeling' funded by the Deutsche Forschungsgemeinschaft (DFG). I was granted a PhD scholarship for my research within this Research Training Group and would like to gratefully acknowledge this financial support.

I wish to thank Hans-Ulrich Klein, now Harvard Medical School/Brigham and Women's Hospital, Boston, and formerly University of Münster, Münster, for our fruitful collaboration for years in the analysis of genomic and epigenomic data, in which I feel we have complemented one another very well. I also thank Martin Dugas, his former group leader at the University of Münster, for supporting our work.

I thank my colleagues for numerous discussions on statistics. Representing many others, in particular I mention Jakob Wieczorek, with whom I shared an office at the Faculty of Statistics, TU Dortmund University, and Bernd Bischl, with whom I shared many coffees during conversations there in the last years.

I thank Holger Schwender, Philipp Berger, Julia Schiffner, Sabrina Herrmann, Claudia Köllmann, Lars Koppers and Jenny Peplies for proofreading this thesis.

Further thanks go to Eva Brune and Brigitte Kupiec for administrative support.

Equally important as work-related aspects, my friends have been providing support in many other ways. Thank you!

I am deeply grateful to Jutta and Thomas as I could not imagine better parents ... and they have made me what I am to a great extent. Thomas also gave me the idea to study statistics by sharing his experience about this exciting profession with me.

Jenny has always been a wonderful big sister and friend, and I am glad to have her!

Finally, I thank Tamara for patience and emotional support during my PhD time... gracias mi vida... eres maravillosa y te amo!

Contents

Chapter 1

Introduction and Motivation

Karl Pearson, sometimes referred to as one of the founders of mathematical statistics, introduced the use of finite mixture distributions as early as 1894. In that year, he fitted a mixture of two univariate normal components to crab measurements based on moments (Pearson, 1894). However, probably only since the introduction of the Expectation-Maximization (EM) algorithm (Dempster *et al.*, 1977), which considerably simplified the fitting of mixture models via the maximum likelihood principle, are such models widely employed for statistical modeling. Their popularity is due to the fact that they arise naturally when a randomly mixed population consisting of K subgroups with relative group sizes $\pi_1^*, \ldots, \pi_K^*$ is to be analyzed. In this frequently occuring situation, inference is to be done on a random variable Y that takes values in $\mathcal{Y} \subseteq \mathbb{R}^\mathfrak{d}$ and is homogenous within the subgroups, but heterogenous across them. This random variable Y describes a characteristic of interest, e.g., physical measurements such as the ratio of forehead to body length in the case of the crab data analyzed by Karl Pearson, or a score summarizing genomic variation that is indicative of the likelihood of a gene to contribute to cancer development. By nature, Y has distinct probability distributions $p(y|\boldsymbol{\theta}_T)$, where $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K)'$ is the (multidimensional) parameter and T is the indicator variable assuming one of the group labels $k = 1, \ldots, K$. The $p(y|\boldsymbol{\theta}_T)$ might be arbitrary parametric

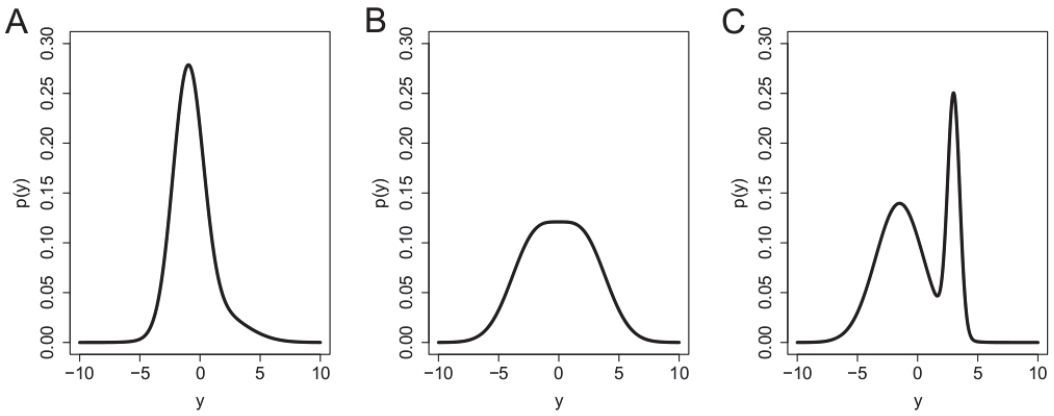

Figure 1.1: Densities of three mixtures consisting of two univariate normal distributions each (Figure **A**: $\mu_1 = -1, \mu_2 = 1, \sigma_1^2 = 1.25, \sigma_2^2 = 2.5, \pi_1^* = 0.8$; Figure **B**: $\mu_1 = -2, \mu_2 = 2, \sigma_1^2 = \sigma_2^2 = 2, \pi_1^* = 0.5$; Figure **C**: $\mu_1 = -1.5, \mu_2 = 3, \sigma_1^2 = 2, \sigma_2^2 = 0.5, \pi_1^* = 0.7$).

densities, but are usually chosen from the exponential family. Mostly, they are assumed to arise from the same parametric family with only $\boldsymbol{\theta}_T$ differing across the groups. In case of $p(y|\boldsymbol{\theta}_T)$ from different parametric families, the $\boldsymbol{\theta}_T$ may be of different dimensions.

An important question is whether the group indicator T can be recorded or not in addition to Y (Frühwirth-Schnatter, 2006). If it can, the probability of sampling from subgroup T is π_T^*, and Y given T follows $p(y|\boldsymbol{\theta}_T)$. The joint density $p(y, T)$ can then be written as

$$p(y, T) = p(T) \cdot p(y|T) = \pi_T^* \cdot p(y|\boldsymbol{\theta}_T).$$

If, however, the group indicator T cannot be recorded in addition to Y, a finite mixture distribution arises. In this typical situation, the marginal density $p(y)$ is given by

$$p(y|\boldsymbol{\theta}, \boldsymbol{\pi}^*) = \pi_1^* p(y|\boldsymbol{\theta}_1) + \ldots + \pi_K^* p(y|\boldsymbol{\theta}_K) \tag{1.1}$$

with weight vector $\boldsymbol{\pi}^* = (\pi_1^*, \ldots, \pi_K^*)'$.

In Figure 1.1, it is visualized how a mixture consisting of just two normal components leads to a considerable increase in flexibility compared to a single

normal density: it may present skewness, tails of different thickness or even multimodality.

There are several advantages of analyzing mixture models in a Bayesian framework, e.g., an increased capability to cope with small sample sizes and a smaller susceptibility to problematic likelihood shapes such as local maxima or unboundedness (for a detailed discussion, see Frühwirth-Schnatter, 2006). In addition, finite Bayesian mixture models have a straightforward generalization to an infinite number of mixture components, leading to nonparametric Bayesian analyses (Ferguson, 1973; Lo, 1984).

There are many situations and applications in which the group indicator T cannot be recorded – i.e. in which a finite mixture distribution as in (1.1) arises – but the mixture distribution is not of primary interest. Instead, the focus lies on the level of each observation, in particular on the allocation of observations to the mixture components (i.e. on the values of T) and measures derived from this allocation. Applications of this kind can be found, e.g., in cluster and classification analyses (throughout this thesis, the term classification will refer exclusively to supervised learning in the sense of discrimination). In this thesis, Bayesian methods employing mixture distributions are described and developed for applications which bear characteristics of both of these research tasks. These procedures perform classification in the sense of allocating objects to previously well-defined classes, but they are similar to clustering in the sense that they are unsupervised, i.e. no training data are available on which the allocation of objects could be learned. Furthermore, in the applications motivating the work presented in this thesis, there are, in principle, no covariates available which could be used to build a classification model. The objects for which a sample of Y is observed in this thesis are proteins in a cell (Chapter 4) or genomic locations (Chapter 5), i.e. points (or intervals) in either $\mathbb{R}^2$ or $\mathbb{R}$. The most usual case of a mixture of normal distributions is considered in Section 5.3. However, a greater focus lies on mixtures of gamma distributions (Section 4.4) and mixtures of normal and either exponential or gamma distributions (Section

5.4), i.e. on non-standard cases in which the probability distributions in the groups may come from different parametric families. Bayesian nonparametrics, which consider an infinite number of components $K \in \mathbb{N}$, are not a focus and only discussed as a possible alternative.

In a project of the former Research Training Group 1032/2 'Statistical Modeling' at the Faculty of Statistics of the TU Dortmund University, spatial patterns of Ras proteins on the plasma membrane were investigated in collaboration with the Max Planck Institute of Molecular Physiology, Dortmund. Essential results of this project have been published in Schäfer *et al.* (2015). Ras proteins are small and measure only about 2-3 nm in diameter. In the project, a special regard was given to their nanoclustering. Nanoclusters are deemed important in intracellular communication, and by consequence, in the coordination of cell regulation. They might thus play a role in the growth and division of cells as well as in tumor development. For this reason, they are a target of biomedical research. Only in recent years, specific implementations of fluorescence microscopy have been developed that provide the spatial resolution necessary to analyze intact, living cells on the nanoscale. In particular, photo-activated localization microscopy (PALM) and stochastic optical reconstruction microscopy (STORM) (Betzig *et al.*, 2006) now reach this resolution.

A key hypothesis in the research project is that nanoscale Ras clustering takes a characteristic form in cancerous cells. This might allow to identify tumors in tissues, and, more importantly, eventually provide a means to inhibit them. The central biological goal is to assess whether the clustering behavior of Ras proteins differs between distinct experimental conditions, such as healthy vs. tumor cells. A first step towards this goal pursued in this thesis is to quantify key parameters of the clustering behavior, in particular the proportion of proteins forming clusters, the mean cluster size and the mean cluster radius. Estimates of these parameters may be compared across different experimental conditions. In addition to point estimates, the clustering parameters' distributions are also of interest, since they contain more information and enable comparisons on

a better fundament. In this respect, the setting differs from classical cluster analysis focusing on the cluster partition. Simultaneous inference for all three above mentioned key parameters is challenging due to identifiability problems resulting from dependencies between cluster size and cluster radius.

The results of this project are presented in Chapter 4 of this thesis. In particular, a new Bayesian approach called GAMMICS (GAMma Mixtures for Inference on Cluster Structures) is proposed that fits a mixture model to the squared distances between proteins and their second nearest neighbors. Inference for all parameters of interest is possible in this framework by incorporating algorithmic aspects into the model. For validation, an extensive simulation study is carried out. A point process model based on the Matérn cluster process (Matérn, 1986) is developed for this purpose. This point process model, the double Matérn cluster process, accounts both for only a proportion of points to cluster and for clusters on two level. Specifically, in addition to clusters, points may form metaclusters potentially containing clusters as well as single points. The GAMMICS method is compared to the popular DBSCAN algorithm for density-based clustering (Ester *et al.*, 1996) and to a Bayesian framework for model based clustering (Fritsch and Ickstadt, 2009). In addition, a comparison is made with a combination of these two approaches (Argiento *et al.*, 2013) as well as to Ripley's K-function and derivations (see, e.g., Kiskowski *et al.*, 2009).

Another application of finite dimensional mixture models is the integrative analysis of different data inputs from 'omics', i.e. a variety of research fields in molecular biology, such as genomics, transcriptomics, epigenomics and proteomics. These scientific fields have gained huge importance in recent years due to technological advances facilitating their analysis. In the past, the success of high-throughput microarray technology has helped to use omics data to gain a better understanding of mechanisms underlying disease pathogenesis on a molecular basis, e.g., in cancer research. In recent years, the surge of analyses based on sequencing technologies has further propelled the availability of omics data and dramatically increased its resolution. Besides leading to a variety of

univariate research, this development has already brought about integrative, mostly paired, analyses of different omics data types. Aims of these analyses include, in particular, the combination of information as well as the validation of results and conclusions obtained from a single omics data type. A prominent example are joint analyses of copy number and gene transcription data including genomic and transcriptomic observations (see, e.g., Ortiz-Estevez *et al.*, 2011; van Wieringen and van de Wiel, 2009; Schäfer *et al.*, 2009).

Chapter 5 of this thesis is concerned with the integrative analysis of omics data, in particular epigenomic data. It is based on a project conducted in collaboration with the Institute of Medical Informatics at the University of Münster, and essential results have been published in Schäfer *et al.* (2012) and Klein *et al.* (2014). In epigenomics, one focus of interest is the analysis of the influence which modified histones – proteins around which the DNA winds – might have on the activation or inhibition of gene transcription. Knowledge about this influence, in turn, might help to get a better understanding of the development of genetically caused diseases. In particular, histone modifications have been reported to be fundamental to stem cell differentiation as well as to the genesis of cancer (e.g., Baylin and Jones, 2011; Dawson and Kouzarides, 2012).

In cancer research, one goal is to find so-called 'driver' genes or transcripts that contribute causally to cancer development through harmful mutations and a consequently altered behavior, as opposed to others that also present conspicuous behavior in cancer cells, but do so only as a normal reaction to the distorted functioning of other genes ('passengers'). To identify such drivers and discriminate them from passengers, one strategy in epigenomics is to find genomic regions in which histone modifications differ between two relevant biological conditions, e.g., between cancer and normal cells, or between cells bearing a mutation of an epigenetic regulator and wildtype cells. The main focus is usually on detecting epigenetic modifications occurring together with a change in gene transcription, since in this case there is more evidence for a contribution to

the phenotype or to cancer development. Hence, in many studies, genome-wide mRNA and histone modification levels are measured from the same samples. Besides their potential causative role in cancer, genes showing differences in both histone modification and gene transcription data are important for identifying putative therapeutic targets for epigenetic drugs.

In this thesis, integrative approaches are proposed to identify interesting genomic loci based on measures inspired by the externally centered correlation coefficient (Schäfer *et al.*, 2009), a correlation coefficient developed to assess whether differences observed between two collectives are equally directed in two input variables. Besides the identification of drivers, this approach is employed to assess the congruence between different technologies employed to measure histone modifications. Correspondingly, the two input variables are either a histone modification measured with two different technologies technologies (Section 5.3) or a histone modification measured in addition to gene transcription (Section 5.4). First, a Bayesian mixture model with a fixed number of Gaussian mixtures is proposed to investigate such equally directed differences, and is applied to an experimental data set. Second, a more general Bayesian mixture model is developed that reduces the number of necessary components by mixing distributions of different types, improving interpretability. This mixture model is validated in applications to several experimental and simulated data sets. It is also compared to a naive approach based on separate analyses of both input variables.

In the mixture model for Ras proteins, two classes are contemplated (clustered vs. non-clustered proteins), whereas in the omics project three classes are considered (genomic loci with observed differences between two collectives that are either equally or unequally directed in two data inputs, and genomic loci for which such differences cannot be observed in at least one input). In both projects, a standard mixture model is extended to satisfy needs posed by the respective scientific questions: in the Ras project, the model is complemented by an algorithmic post-processing step to carry out further, otherwise untractable

estimations of additional parameters, while in the omics project, classes may be represented by several mixture components and the type of distribution may vary across classes.

This thesis is organized as follows: in Chapter 2, biological and biotechnical background information is presented for the problems tackled in the two main projects, while Chapter 3 gives a theoretical overview on mixture models and a brief description on clustering methods. Chapter 4 is dedicated to the project on the spatial analysis of Ras proteins. Specifically, in Section 4.4, the simulation setup for validating the considered methods is described. Subsequently, the GAMMICS method and the considered competitors are described. The performance of these methods on the simulated and experimental data is reported in Section 4.7. The focus of Chapter 5 lies on the integrative omics project: Section 5.2 considers the adaptation of the externally centered correlation coefficient. Approaches for the integrated analysis of sequencing-based histone modification measurements and another omics input are described in Section 5.3 and Section 5.4 along with results, respectively. Finally, the results of the analyses are summarized and discussed in Chapter 6. In the Appendix, supplementary plots and tables are displayed, MATLAB, `R` and BUGS code for methods proposed in this thesis are documented, and additional theoretical considerations omitted in the main text are given.

Chapter 2

Background

In this chapter, first the biological background on genetic and epigenetic measurements is discussed, followed by a detailed description of the biology of Ras proteins. The order of these topics reflects the fact that proteins arise as a product of gene transcription. For a more detailed introduction to molecular genetics, see, e.g., Strachan and Read (1999).

2.1 Genetic and Epigenetic Measurements

All living organisms consist of basic units called *cells*. They contain fundamental elements of the respective organisms, such as the genomic information, and are delimited by a *plasma membrane*. The *genome* of an organism has the role of coding its structure and its functionality at the cellular level. It is stored by means of the *deoxyribonucleic acid (DNA)*, which in eucaryotes (organisms whose cells contain a nucleus) mostly coils to form *chromosomes*. Humans, as well as most animals, have a diploid set of chromosomes, i.e. every chromosome is available in two copies, where one copy comes from each parent. The number of chromosome pairs varies between species. Humans, e.g., have 23 and house mice (in Latin: mus musculus) 20 pairs of chromosomes in the nucleus of every *somatic* cell. Somatic cells (from Greek 'soma'= body) are all cells which are not *germ cells* or *gametes* (responsible for sexual reproduction) or *stem cells*.

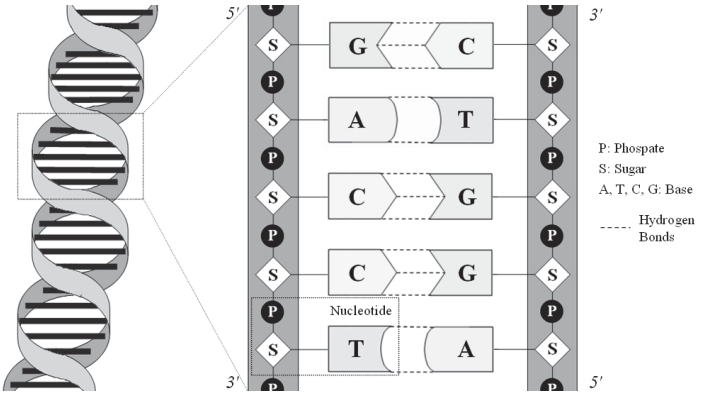

Figure 2.1: Structural composition of the DNA. Its two spiral strands are long chains of nucleotide molecules, each nucleotide consisting of three elements: phosphate (P), the sugar (S) deoxyribose and one of the four bases adenine (A), cytosine (C), guanine (G), and thymine (T). Bases lying on opposed strands at the same position are connected via hydrogen bonds, where adenine always binds to thymine and cytosine always binds to guanine (key-lock-principle). Source: Schwender (2007).

The DNA has the form of a double helix, i.e. two intertwined spiral strands consisting of long chains of *nucleotides* (see Figure 2.1). A nucleotide is a molecule consisting of phosphate, the sugar deoxyribose and one of the four bases adenine (A), cytosine (C), guanine (G), and thymine (T). Bases lying on opposite sides at the same DNA location are connected via hydrogen bonds, where adenine always binds to thymine and cytosine always binds to guanine. Due to this key-lock-principle of the bases known as complementary base-pairing, it is sufficient to know the nucleotide sequence of just one of the two strands in order to possess the full coded information. The length of DNA fragments is usually measured in 'base pairs' (bp). Due to conventions regarding the naming of carbon atoms in the nucleotides on a chemical level, genomic loci are usually referred to relative to each other as 'towards the 3 prime end' (downstream, short: 3' end) or 'towards the 5 prime end' (upstream, short: 5' end) on one of the DNA strands.

Besides the DNA, the chromosomes also contain proteins, most of which are *histones* around which the DNA winds. The formation of this DNA-protein complex, jointly referred to as *chromatin*, leads to a more compact packing of

the DNA, guaranteeing that it fits into the cellular nucleus.

Structurally, the information stored in the DNA consists of *genes*, small sections each containing the blueprint for one or more proteins.

Proteins are fundamental components of cells responsible for virtually all body functions of living organisms. The central dogma of molecular biology (see Figure 2.2) describes how the information coded in genes is used to form proteins: first, the DNA is transcribed into *messenger ribonucleid acid (mRNA)* by means of the enzyme *RNA polymerase*. The mRNA is similar to the DNA, but consists only of one strand and contains a different sugar molecule (ribose instead of deoxyribose) as well as a different base (uracil instead of thymine). After this *transcription*, the mRNA leaves the cell nucleus. The mRNA product of a gene is called *transcript*. During *translation*, sequences of three consecutive nucleotides of the mRNA are translated into one of 22 *amino acids* forming a chain that coils into a protein.

The amount of specific mRNA detected for a specific organism and DNA locus is referred to as *(gene) transcription level*. The term *gene expression* (GE) is often used synonymously to gene transcription, but sometimes refers to the entire process that forms a protein using the information coded in the DNA sequence of a gene. To avoid misunderstandings, the term gene transcription is generally used in this thesis (except for the commonly used term 'gene expression microarray'). The position at which the transcription begins for a specific gene is called its *transcriptional start site (TSS)*.

Near the TSS of each gene, there exists a DNA region called *promoter* that helps to direct the RNA polymerase to the correct position in order to begin transcription of this gene. To do this, it interacts with DNA binding proteins called *transcription factors*. If a gene codes for more than one protein, there will consequently be several promoters and one speaks of *alternative promoter usage*.

Structurally, genes can be viewed as a sequence of *introns* and *exons*. Introns (short for intragenic regions) are DNA regions that are not translated

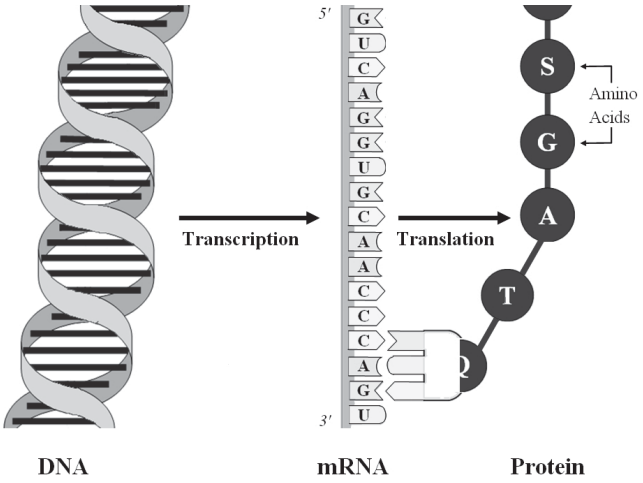

Figure 2.2: The central dogma of molecular biology explains how proteins are formed based on the information coded in the DNA: The DNA is first transcribed into single-stranded messenger ribonucleid acid (mRNA) by means of the enzyme RNA polymerase. Subsequently, sequences of three consecutive nucleotides of the mRNA are translated into one of 22 amino acids, forming a chain that coils into a protein. Source: Schwender (2007).

into proteins, they are removed during a process called *splicing*. The remaining DNA parts, the information-bearing exons (short for expressed regions), eventually do code for proteins. However, for most genes the delimitations between introns and exons are defined only during splicing, resulting in several different proteins generated from the same gene. This is called *alternative splicing* and, according to estimates, affects approximately 60% of genes in humans (Chen, 2011). To take this into account, transcripts will be considered as elementary units instead of genes in Section 5.4 of this thesis.

The importance of genomic analyses in cancer research is owed to the fact that the DNA can vary between individuals of the same species. Even though most DNA is identical throughout a species, existing genetic variation may lead to different proteins, and hence to different risks for diseases such as cancer.

Genetic variation may arise in two different ways. One the one hand, genes from both parents are regularly rearranged to form a new genome in *recombination* during reproduction. One the other hand, *mutations* may occur, resulting in completely new characteristics that neither parent possessed. While muta-

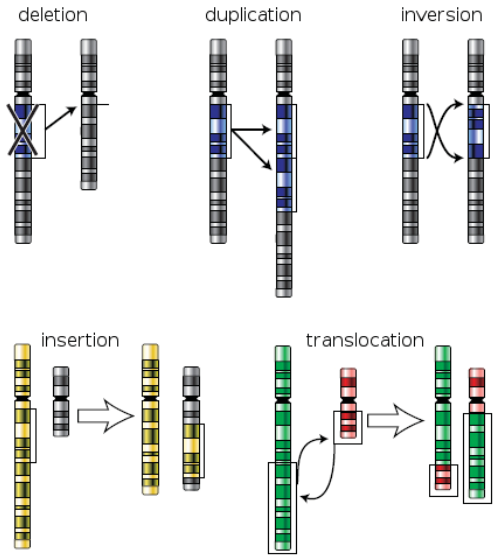

Figure 2.3: Possible chromosome mutations in humans. Two chromosomes are involved in insertions and translocations, while only one chromosome is involved in the other mutations. Source: Wikipedia (2014a), slightly modified.

tions are an integral part of evolution just as recombination, in many cases they lead to harmful rather than beneficial alterations of the genome and for this reason are frequently involved in the development of (genetic) diseases. When they occur in germline cells, they may be passed on to the next generation. If they occur in somatic cells, however, they only affect the organism in which they occur. In the context of investigating genetic causes of malign tumors, mostly somatic cells affected by cancer are examined. The possible occurrences of a gene, or, more general, of a specific sequence of nucelotides on the DNA, are called *alleles*.

The number of copies of such an allele in the genome is generally referred to as *DNA copy number* or just *copy number* (CN). In case of a diploid chromosome set, each of the two copies of the chromosome to which a specific DNA sequence belongs typically carries an identical copy of the sequence. This corresponds to a copy number of two, although exceptions exist, e.g., in case of the X- and Y-chromosome in males. Since the copy number refers to specific alleles, measurements of the copy number always refer to a reference genome, usually

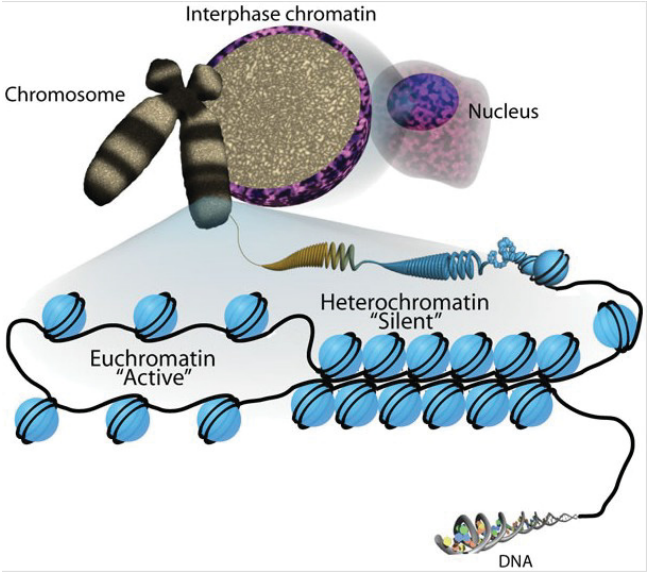

Figure 2.4: Chromatin organization. Euchromatin (loose or open chromatin) structure is permissible for transcription and corresponds to histone acetylation. Heterochromatin (tight or closed chromatin) structure is inhibitive for transcription and corresponds to histone methylation. Source: Sha and Boyer (2009).

the wild type, i.e. the most typical or frequent allele.

Mutations may affect DNA regions of very different sizes, from a single base to whole parts of chromosomes, and lead to structural alterations in these regions.

If a new DNA sequence variation becomes increasingly common in the genomic pool, i.e. when the most infrequent allele in a DNA sequence reaches a frequency of 1% throughout a population, it is called a polymorphism. When the variation affects only a single nucleotide, i.e. only a single base is altered, deleted, or added, it is referred to as a *single nucleotide polymorphism (SNP)*.

Mutations of whole parts of a chromosome are caused by larger deletions, duplications, inversions, insertions, and translocations (see Figure 2.3) and may have a strong impact depending on their position. Translocations may be written in short in the form t(1;2) (denoting a translocation between chromosome 1 and chromosome 2).

There are other factors besides the DNA that have an impact on the gen-

eration of proteins. These are usually referred to as epigenetic influences. One of these epigenetic influences is the modification of histones that may have either a repressive or an activating effect on gene transcription. On a chemical level, *methylation* of histones generally results in a dense chromatin structure, inhibiting gene transcription, whereas *acetylation* of histones leads to a loose chromatin structure that activates gene transcription (see Figure 2.4). Investigating the levels of methylation and acetylation may, thus, shed light on the extent to which gene transcription is being influenced epigenetically. Histone modifications are fundamental to stem cell differentiation as well as to the genesis of cancer (Baylin and Jones, 2011; Dawson and Kouzarides, 2012). Analyzing their functionality may, therefore, help to understand the cause of genetic diseases.

A central step in the measurement of omics data such as DNA copy number, gene transcription and histone modifications is the quantification of DNA, mRNA, or DNA bound to histones, respectively. This implies the identification of the DNA fragments. For this procedure, inherent to analyses of all three mentioned omics data types, several technologies have been proposed and constantly advanced regarding their throughput and resolution. The DNA fragments may be charged with fluorescent dye and hybridized to a microarray containing millions of small fixed DNA sequences, on which the relative amount of hybridized DNA may be measured with a scanner. Alternatively, the DNA fragments may be sequenced directly using next-generation sequencing technologies and compared to a reference genome. Sequencing technologies provide maximum resolution and sensitivity without restriction to a fixed number of probes. Both microarrays and sequencing methods are described in more detail in the Appendix A.1.

The measurement of histone modifications requires an additional pre-step, since not only amounts of DNA or mRNA, but interactions between DNA and proteins need to be measured. The standard method to approach such DNA-protein complexes is *chromatin immunoprecipitation (ChIP)*, a molecular bi-

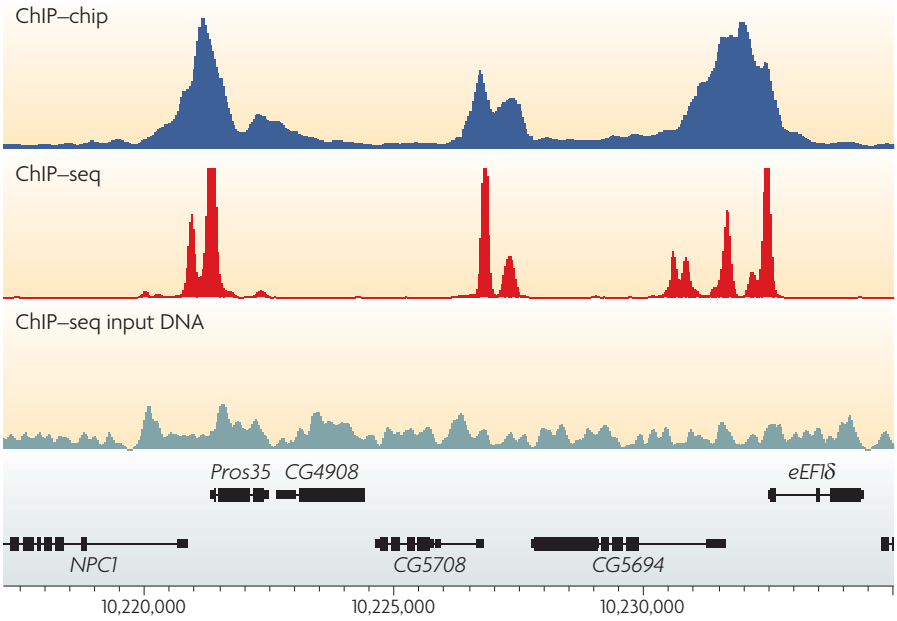

Figure 2.5: Example profiles produced by chromatin immunoprecipitation followed by sequencing (ChIP–seq tag density; red) or microarray analysis (unlogged ChIP–chip intensity ratio; blue). Shown are binding profiles of the chromodomain protein Chromator in a Drosophila melanogaster cell line. The ChIP–seq profile reveals Chromator binding positions with higher spatial resolution and sensitivity. The ChIP–seq input DNA (control experiment) tag density is shown in grey for comparison. Source: Park (2009).

ologic method for investigating bindings between DNA and proteins such as modified histones. In ChIP, DNA bound to histone proteins is sheared into small fragments in this context. Then, modified histones with DNA fragments wound around them are immunoprecipitated (i.e. isolated) by means of an antibody specific to the histones of interest. Afterwards, the genetic material is separated from the histones and quantified (Huss, 2010).

As an example, in Figure 2.5 ChIP binding profiles of the so-called Chromator protein are compared resulting from analyses based on microarray (ChIP-chip; Mockler and Ecker, 2005) and sequencing technology (ChIP-seq; Park, 2009; Furey, 2012). This figure shows that the ChIP–seq profile has a higher spatial resolution and sensitivity, and the identified peaks – indicative of a signal clearly distinguishable from noise – are sharper and narrower. A ChIP–seq control profile is also visualized which unsurprisingly does not exhibit any peaks.

The genomic locations obtained by sequencing technologies (ChIP-seq, RNA-seq, or other applications) result in count data as raw values when analyses are performed on a gene-wise level. Thus, a distinct preprocessing is required than for microarray experiments which render continuous raw values.

Microarray measurements have to be normalized to remove unwanted effects like noise and artifacts in the signal. The specific normalization approach adopted in this thesis is discussed in Sections 5.3.2 and 5.4.2.

In sequencing experiments, the absolute number of DNA fragments found for a specific genomic location (also called *read count*) depends on the *sequencing depth* or *coverage level*, i.e. the number of times a genome is sequenced. The sequencing depth is arbitrarily chosen by the analyst according to a tradeoff: a higher sequencing depth leads to a higher sensitivity, but also to higher costs. To avoid an influence of the sequencing depth on the results of an experiment, either the absolute number of reads has to be adjusted for different sequencing depths across data sets by a normalization procedure, or a quantification other than the absolute number of reads has to be chosen. Specific methods employed in this thesis are discussed in Section A.9 of the Appendix as well as in Sections 5.3.2 and 5.4.2. Normalization approaches are often referred to as methods for the estimation of transcript abundance.

Specific methods for alignment of sequences and normalization are not discussed in this thesis.

2.2 Ras Proteins

In Chapter 4, the spatial alignment of certain proteins on the plasma membrane of cells is of interest due to their role in intracellular communication. Cellular signal transduction plays an important role in cell regulation, i.e. in processes intended to maintain a balanced, healthy cell condition. Among other issues, this involves the regulation of the growth of a cell and its replication rate, both investigated in cancer research since tumors are a product of uncontrolled

growth and replication.

The communication between cells takes place by means of *receptors*, i.e. specific proteins located on the cell surface or beneath it that bind chemical signals. Signals reaching the cell from outside through the receptors may lead to a variety of events within the cell, eventually activating mechanisms changing its behavior in a *cellular response* to the signal. The cellular responses depend on chemical and physical properties of the cells as well as on spatial inhomogeneities in the distribution of involved signaling proteins. Specifically, it is assumed that the spatial patterns of such proteins on the plasma membrane have an influence on their signaling behavior, and protein clusters occurring at a sub-micrometer scale are believed to be of relevance. However, it is still largely unknown how these clusters arise and what specific function they have in the signal transduction (Jacobson *et al.*, 2007).

The signaling chain of the epidermal growth factor (EGFR) on the plasma membrane, fulfilling an obvious role in cell regulation through its influence on cell growth, is regarded as an emblematic case of such protein clustering (Citri and Yarden, 2006). In particular, this signaling chain includes ErbB-1, the receptor of the epidermal growth factor, and subsequent steps belonging to the regulatory circuit of the protein *GTPase Ras*. Both of these molecules have key functions in signal transduction involved in processes related to cell development and growth. Such processes include maintaining the integrity of the cellular skeleton, cell differentiation, proliferation, and the controlled cell death. Thus, these molecules can play a major role in the development of tumors and are therefore a target for biomedical research. Both proteins also do partly cluster on a nanometer scale (Hancock, 2006; Saffarian *et al.*, 2007). The work in this thesis focuses on the spatial clustering of Ras proteins, 2-3 nm in diameter, which have been given special research attention (Henis *et al.*, 2009; Prior and Hancock, 2012). Their name reflects their first discovery in the context of the investigation of rat sarcomas (sarcomas are malign tumors originating in connective tissues).

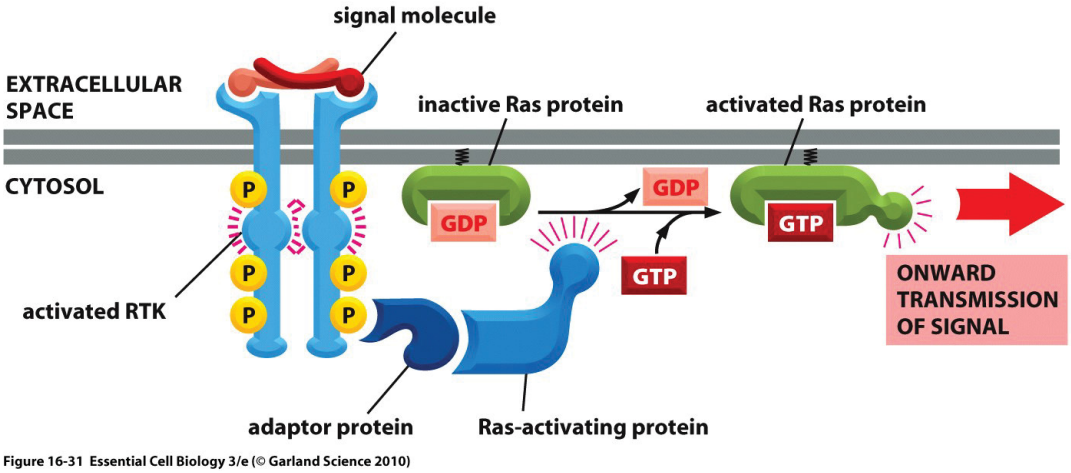

Figure 2.6: Signal transduction via Ras proteins: Signal molecules bind to (and activate) receptor tyrosine kinases (RTK), enzyme-coupled cell-surface receptors. An adaptor protein docks on a phosphotyrosine (P) on the activated receptor and recruits a Ras-activating protein that causes Ras to exchange its bound guanosine diphosphate (GDP) for guanosine triphosphate (GTP). The activated Ras protein can now stimulate several downstream signaling pathways. Source: Alberts *et al.* (2010).

The Ras protein family has various members differing with respect to their chemical composition, in particular concerning the part anchoring them to the membrane. Two prominent examples are H-Ras and K-Ras.

Alongside many other proteins involved in propagating arriving signals further into the cell, Ras proteins are guanosine nucleotide-binding proteins, sometimes referred to as binary molecular switches due to their two possible states. On the chemical level, such proteins may either bind to guanosine triphosphate (GTP) or guanosine diphosphate (GDP). Binding to GTP results in an active state, while binding to GDP leads to inactivity. In the literature, proportions of 30-45% of Ras proteins have been reported to form highly dynamic, short-lived nanoclusters having an average radius of 6-16 nm and an average cluster size of 6-7 proteins (Plowman *et al.*, 2005; Tian *et al.*, 2007; Kiskowski *et al.*, 2009). However, the methodology in these publications may be partly questioned from a statistical point of view, as will be discussed in Section 4.4.1. The clusters are formed by Ras during its inactive state and are assumed to be stabilized during

the active GTP-bound state. This results in the formation of signaling platforms possessing characteristics beneficial for the transmission of signals into the cell (Tian *et al.*, 2007; Rotblat *et al.*, 2010). This process, symbolized in Figure 2.6, eventually influences gene transcription involved in cell proliferation and other cancer-related regulations via a subsequent protein cascade. It has been demonstrated that inhibition of Ras clustering severely affects signal transduction (Plowman *et al.*, 2005). Moreover, Ras proteins deregulated as a consequence of mutations have been found in cancers (Fernández-Medarde and Santos, 2011; McCubrey *et al.*, 2012).

The direct visualization of small clusters formed by signaling proteins such as Ras, in the order of less than 10 nm, has been a challenge until recently. Electron microscopy has traditionally been used to visualize the distribution of Ras proteins (Plowman *et al.*, 2005; Hancock, 2006), but it requires destructive preparation steps for the cellular material that can considerably affect the spatial patterns of interest. Fluorescence microscopy is a newer, alternative technology allowing to examine intact, living cells. However, it has to be combined with special methodologies to reach the necessary resolution. Indirect methods such as fluorescence resonance energy transfer (FRET microscopy) or fluorescence correlation spectroscopy (FCS) have been tested for this task, but these procedures are based on many physical assumptions and are not able to visualize the spatial protein distribution directly. Most recently, the development of photoactivated localization microscopy (PALM) and stochastic optical reconstruction microscopy (STORM) has made it possible to visualize single molecules at an increased spatial resolution of roughly 20 nm (Betzig *et al.*, 2006).

In this thesis, a focus lies on the analysis of data generated with PALM in a setup briefly described in the following. Fluorescence microscopy in general is based on the excitation of fluorescent probes and the subsequent measurement of emitted light. In PALM, appropriate photoactivatable fluorophores such as paGFP, pamCherry, or mEos are genetically fused to the proteins to accomplish

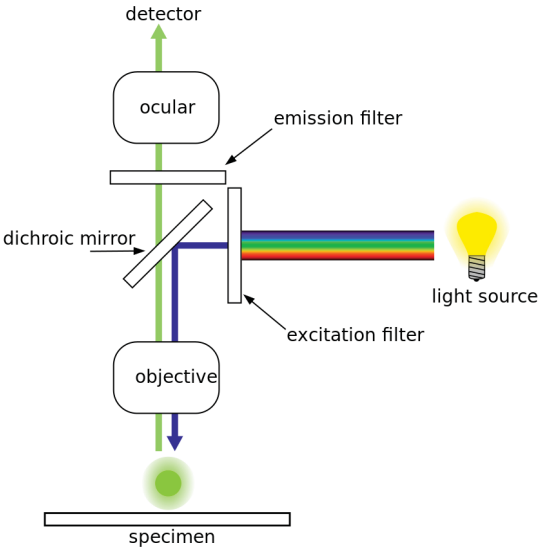

Figure 2.7: Schematic overview of a fluorescence microscopy system. Light emanating from a light source such as a laser is directed at the cell specimen on the glass slide of a microscope by means of a mirror. The photons are absorbed by the specimen and then re-emitted by fluorophores fused genetically to the proteins of interest. An ocular or a digital camera connected to the microscope may then capture the light emitted by a single protein as a blob that covers a surface much bigger than the protein itself. Source: Wikipedia (2014b).

this. A laser dispenses light directed at the cell specimen on the glass slide of a microscope. The photons are absorbed and then re-emitted by the fluorophores, producing light that covers a surface much bigger than the proteins themselves. When instead of an ocular, a digital camera is connected to the microscope, this light can later be seen on resulting greyscale pixel images as blobs indicating the presence of a protein. The images analyzed here have a resolution of 512x512 pixels, where each pixel has a side length of 107 nm. Figure 2.7 shows the principal setup of a comprehensive fluorescence microscopy system in a schematic way.

The research effort presented in this thesis focuses on cells with an expression level above normal typical for cancer cells. In the experimental setup, cancer cells are mimicked by artificially overexpressed Ras, tagging it to a fluorescent protein. Due to the high expression level, not all proteins can be visualized and

detected as single molecules simultaneously because the corresponding blobs would overlap and render a completely white image. Thus, only a subset of molecules are activated by UV light at a time (Hess *et al.*, 2006), avoiding too much overlap between the light blobs. After a reasonable time during which pictures are created every 100 milliseconds, it can be expected that the recorded blobs represent a vast majority of Ras proteins in the cell. The time necessary to record all proteins cannot be determined with certainty. Blobs may persist during several time frames recorded by the camera before they disappear. In some cases, proteins may be reactivated later ('blinking').

In an effort to limit sources of noise, the total internal reflection fluorescence technique (TIRF, Toomre and Manstein, 2001) is employed in combination with PALM. This ensures that only fluorophores in the basal membrane (up to 100-300 nm below the cell surface) are activated, while the fluorophores located deeper inside the cell, of no interest for the cluster phenomenons under investigation, are ignored.

Chapter 3

Mixtures and Clustering

This chapter starts with an introduction to the theory of finite Bayesian mixtures in Section 3.1. Subsequently, Section 3.2 contains brief descriptions of the principal techniques for fitting mixture models, i.e. the Expectation-Minimization (EM) algorithm and Markov Chain Monte Carlo (MCMC) algorithms. Parts of these first two sections follow roughly Section 2.1 in Fritsch (2010). Finally, in Section 3.3, methods for cluster analysis are briefly reviewed with an emphasis on model-based clustering.

3.1 Bayesian Mixtures

3.1.1 Basic Definition

As discussed in the introduction, a finite mixture distribution arises for a random variable Y assuming values in $\mathcal{Y} \subseteq \mathbb{R}^{\mathfrak{d}}$ when its marginal density $p(y)$ is given as a weighted sum over probability densities $p(y|\theta_k)$, i.e. as

$$p(y|\boldsymbol{\theta}, \boldsymbol{\pi}^*) = \sum_{k=1}^{K} \pi_k^* p(y|\theta_k) \tag{3.1}$$

where it is $\pi_k^* \geq 0$ for $k = 1, \ldots, K$, and $\sum_{k=1}^{K} \pi_k^* = 1$, with parameter vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_K)'$ and weight vector $\boldsymbol{\pi}^* = (\pi_1^*, \ldots, \pi_K^*)'$. For the discussion in

this section, K may also be infinite. Existence of densities with respect to the Lebesgue measure will henceforth be assumed.

There are many applications, e.g., cluster and classification analyses, in which the allocation of observations to the mixture components and measures derived from this allocation are of interest rather than the mixture distribution itself. In these cases, the model (3.1) can be reformulated to account for these research focuses by introducing an extra variable T that represents the allocations to the mixture components. Thus, $\pi_k^* = P(T = k|\boldsymbol{\pi}^*)$ and $p(y|\theta_k) = p(y|T = k, \boldsymbol{\theta})$ so that (3.1) becomes

$$p(y|\boldsymbol{\theta}, \boldsymbol{\pi}^*) = \sum_{k=1}^{K} P(T = k|\boldsymbol{\pi}^*)p(y|T = k, \boldsymbol{\theta}).$$

In classification analysis, $T = k$ represents a classification of an object to class/component k, while in cluster analysis, it stands for an allocation to cluster k.

A mixture model estimates $\boldsymbol{\pi}^*$ and $\boldsymbol{\theta}$ as basis for classifying objects. In a situation in which they are known, the conditional distribution of $T|y$ can be obtained by Bayes' theorem, i.e. by

$$p(T = \tilde{k}|y, \boldsymbol{\theta}, \boldsymbol{\pi}^*) = \frac{\pi_{\tilde{k}}^* p(y|\theta_{\tilde{k}})}{\sum_{k=1}^{K} \pi_k^* p(y|\theta_k)}, \tag{3.2}$$

and an object is classified by assigning it to the class $\tilde{k}$ that maximizes (3.2).

Considering realizations $\boldsymbol{y} = (y_1, \ldots, y_n)'$ of independent random variables $Y_1, \ldots, Y_n$ for which the general mixture model (3.1) is assumed, the likelihood function can be written as

$$p(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{\pi}^*) = \prod_{i=1}^{n} p(y_i|\boldsymbol{\theta}, \boldsymbol{\pi}^*) = \prod_{i=1}^{n} \left(\sum_{k=1}^{K} \pi_k^* p(y_i|\theta_k) \right). \tag{3.3}$$

If, again, the focus is on the individual allocations $\boldsymbol{T} = (T_1, \ldots, T_n)'$, (3.3) can be written in an alternative form as well, replacing π_k^* by the indicator function

$\mathbf{I}(T_i = k)$, leading to

$$p(\boldsymbol{y}|\boldsymbol{T}, \boldsymbol{\theta}) = \prod_{i=1}^{n} \left(\sum_{k=1}^{K} \mathbf{I}(T_i = k) p(y_i|\theta_k) \right) = \prod_{i=1}^{n} \left(\prod_{k=1}^{K} p(y_i|\theta_k)^{\mathbf{I}(T_i=k)} \right).$$

In the simplest case of $K = 2$, considered in Chapter 4, $\boldsymbol{\pi^*} = (\pi^*, 1 - \pi^*)'$. In this case, $T_i \sim \text{Bernoulli}(\pi^*)$ is a reasonable assumption, i.e.

$$\begin{aligned} p(\boldsymbol{T}|\boldsymbol{\pi^*}) &= \prod_{i=1}^{n} (\mathbf{I}(T_i = 1)\pi^* + \mathbf{I}(T_i = 2)(1 - \pi^*)) \\ &= \prod_{i=1}^{n} \left({\pi^*}^{\mathbf{I}(T_i=1)} \cdot (1 - \pi^*)^{\mathbf{I}(T_i=2)} \right). \end{aligned} \tag{3.4}$$

For notational consistence, let the possible realizations of the T_i be 1 and 2 in this section, instead of the common values 0 and 1. For a general $K > 2$, it is assumed that $T_i \sim \text{Categorical}(\pi_1^*, \ldots, \pi_K^*)$, i.e. the allocation variable T_i follows a categorical distribution,

$$\begin{aligned} p(\boldsymbol{T}|\boldsymbol{\pi^*}) &= \prod_{i=1}^{n} \left(\sum_{k=1}^{K} \mathbf{I}(T_i = k)\pi_k^* \right) \\ &= \prod_{i=1}^{n} \left(\prod_{k=1}^{K} {\pi_k^*}^{\mathbf{I}(T_i=k)} \right) = \prod_{k=1}^{K} {\pi_k^*}^{\sum_{i=1}^{n} \mathbf{I}(T_i=k)}. \end{aligned} \tag{3.5}$$

The categorical distribution, thus, is a generalization of the Bernoulli distribution for a categorical variable with more than two possible outcomes. The likelihood (3.3) in this notation is obtained as

$$p(\boldsymbol{y}|\boldsymbol{T}, \boldsymbol{\theta}) = \sum_{\boldsymbol{T} \in \{1,\ldots,K\}^n} p(\boldsymbol{y}|\boldsymbol{T}, \boldsymbol{\theta}) \cdot p(\boldsymbol{T}|\boldsymbol{\pi^*}).$$

For notational convenience, the abbreviation $n_k := \sum_{i=1}^{n} \mathbf{I}(T_i = k)$ will be used in the remainder of this thesis for the number of observations allocated to component k.

3.1.2 Prior Distributions

In a Bayesian framework, it is necessary to specify prior distributions for the model parameters. The parameters $\boldsymbol{\theta}$ and $\boldsymbol{\pi}^*$ are usually assumed to be independent a priori so that

$$p(\boldsymbol{\theta}, \boldsymbol{\pi}^*) = p(\boldsymbol{\theta})p(\boldsymbol{\pi}^*).$$

The specification of the mixture prior mixture $p(\boldsymbol{\pi}^*)$ will be discussed with particular attention. In the simplest case of $K = 2$, where $\boldsymbol{\pi}^* = (\pi^*, 1 - \pi^*)'$, a Beta prior with parameters $a_{\pi^*}, b_{\pi^*} > 0$ is assumed for π_1^* with density

$$p(\pi^*) = \frac{\Gamma(a_{\pi^*})\Gamma(b_{\pi^*})}{\Gamma(a_{\pi^*} + b_{\pi^*})}\pi^{*a_{\pi^*}-1}(1-\pi^*)^{b_{\pi^*}-1} \propto \pi^{*a_{\pi^*}-1}(1-\pi^*)^{b_{\pi^*}-1} \cdot \mathbf{I}(\pi^* \in [0,1]).$$

The posterior distribution with density

$$p(\pi^*|\boldsymbol{T}) \propto \pi^{*a_{\pi^*}+n_1-1}(1-\pi^*)^{b_{\pi^*}+n_2-1}$$

is a Beta$(a_{\pi^*} + n_1, b_{\pi^*} + n_2)$ distribution, i.e. it has the same functional form as the prior. The Beta distribution is thus by definition the natural conjugate distribution for the Bernoulli distribution of T_i in (3.4). When $K > 2$, the standard prior for $\boldsymbol{\pi}^* = (\pi_1^*, \ldots, \pi_{K-1}^*, 1 - \sum_k^{K-1} \pi_k^*)'$ is a Dirichlet distribution with parameter $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)', \alpha_k > 0$ for $k = 1, \ldots, K$. This distribution has the density

$$p(\boldsymbol{\pi}^*) = \prod_{k=1}^{K} \pi_k^{*\alpha_k - 1} \cdot \frac{\Gamma(\sum_{k=1}^K \alpha_k)}{\prod_{k=1}^K \Gamma(\alpha_k)} \propto \prod_{k=1}^{K} \pi_k^{*\alpha_k - 1} \cdot \mathbf{I}(\pi_k^* \in [0,1])$$

and will be denoted by Dirichlet$(\boldsymbol{\alpha})$. Analogously to the case $K = 2$, the Dirichlet distribution is the natural conjugate prior for the categorical distribution of $\boldsymbol{T}$ in (3.5), since the posterior

$$p(\boldsymbol{\pi}^*|\boldsymbol{T}) \propto p(\boldsymbol{T}|\boldsymbol{\pi}^*)p(\boldsymbol{\pi}^*) \propto \prod_{k=1}^{K} \pi_k^{*\alpha_k + n_k - 1} \tag{3.6}$$

is a Dirichlet$(\alpha_1 + n_1, \ldots, \alpha_K + n_K)$.

As reflected by these relationships, the Dirichlet distribution is the multivariate generalization of the Beta distribution. The natural conjugacy (henceforth: conjugacy) of the Beta and Dirichlet distributions, respectively, is a principal motivation for employing them, since choosing conjugate prior distributions in Bayesian statistics always has advantages when fitting the model. Conjugate prior distributions have computational benefits: in simple models they often allow an analytical computation of the posterior distribution, while in more complex models, they facilitate the use of the Gibbs sampler in MCMC approaches (for details, see Section 3.2.2). Conjugate prior distributions also often help to interpret results and have the nice property of being interpretable as additional data (Gelman *et al.*, 2013).

There exist several extensions and generalizations of the Dirichlet distribution. Important examples are the generalized Dirichlet distribution (Ishwaran and James, 2001) or the two-parameter Poisson-Dirichlet distribution (Pitman and Yor, 1997).

The numerous priors for infinite mixture models discussed in the field of nonparametric Bayesian statistics can, at least in their great majority, also be viewed as extensions or generalizations of the Dirichlet distribution. The oldest and most important one is the Dirichlet process (Ferguson, 1973) that induces Dirichlet distributions when data are grouped into any finite measurable partition, exploiting the conjugacy of the Multinomial-Dirichlet model for this setting. The Dirichlet process mixture (DPM) model introduced by Lo (1984) (see also MacEachern and Müller, 1998) is one of the most frequently employed Bayesian mixture models.

For the Dirichlet process in turn, many modifications and generalizations have been developed. For example, several hierarchical extensions have been introduced. While they facilitate to appropriately borrow information across data originating from distinct sources such as different study centers, they also help to take into account variation in the target variable Y over time or due to values

of covariates (Dunson, 2010). These extensions, in particular, include mixtures of independent Dirichlet processes (Müller *et al.*, 2004), dependent Dirichlet processes (McEachern, 1999), hierarchical Dirichlet processes (Teh *et al.*, 2006), and nested Dirichlet processes (Rodriguez, 2008).

Recent developments include latent stick-breaking processes that, in contrast to other nonparametric models, facilitate anisotropic and nonstationary models (Rodriguez *et al.*, 2010). Another recent proposal are probit stick-breaking processes, in which the characteristic beta distribution in the stick-breaking construction, see the definition in (3.8), is replaced by probit transformations of normal random variables (Rodriguez and Dunson, 2011).

Overviews on the rapidly evolving field of Bayesian nonparametrics are given in Ghosh and Ramamoorthi (2003) and Hjort *et al.* (2010). A comprehensive theoretical overview of classes of nonparametric priors providing more flexibility than the Dirichlet process is given, in particular, by Lijoi and Prünster (2010).

A connection between finite and infinite mixture models exists in truncated versions of infinite mixture models such as the truncated Dirichlet process, in which an upper bound is fixed for the number of components K, approximating the infinite by a finite mixture model (see Ishwaran and Zarepour, 2000; Ishwaran and James, 2001). When a high number of components is used, the difference to an infinite mixture becomes negligible, while the dimensionality of the model can be kept fixed, leading to computational benefits (cf. Section 3.2.3).

A general class of both finite and infinite mixture priors is the class of Ongaro-Cattaneo random measures (Ongaro and Cattaneo, 2004):

Definition 3.1 (**Ongaro-Cattaneo probability measures**)**.**
A random discrete probability measure P is called Ongaro-Cattaneo probability measure if it can be written as

$$P = \sum_{k=1}^{K} \pi_k^* \delta_{\zeta_k^*}^*, \tag{3.7}$$

Priors with $K < \infty$	a_k	b_k	Note
Dirichlet distribution	α_k	$\sum_{k^*=k+1}^{K} a_{k^*}$	$k = 1, \ldots, K-1$, set $V_K = 1$
Finite-dimensional Dirichlet	α^*/K	$\sum_{k^*=k+1}^{K} a_{k^*}$	$k = 1, \ldots, K-1$, set $V_K = 1$
Trunc. Dirichlet(α_0) process	1	α_0	set $\pi_K^* = 1 - \sum_{k=1}^{K-1} \pi_k^*$
Priors with $K = \infty$			
Dirichlet(α_0) process	1	α_0	
Pitman-Yor(a,b) process	$1-a$	$b+ka$	$a \in [0,1)$, $b > -a$

Table 3.1: Sampling of known priors via stick-breaking construction. For the Dirichlet distribution, the finite dimensional Dirichlet prior, the truncated Dirichlet process, the Dirichlet process and the Pitman-Yor process, corresponding choices for $\boldsymbol{a} = (a_1, \ldots, a_K)'$ and $\boldsymbol{b} = (b_1, \ldots, b_K)'$ are listed when sampling the proportions $V_k \sim \text{Beta}(a_k, b_k)$ independently. It is $a_k, b_k, \alpha^*, \alpha_0 > 0$.

where K, π_k^*, and $\delta_{\zeta_k^*}^*$ are random variables with the following properties: K is a positive integer or ∞. The weights $\pi_1^*, \ldots, \pi_K^*$, conditional on K, have an arbitrary distribution on the $(K-1)$-dimensional simplex

$$S_M = \left\{ \boldsymbol{\pi^*} \in \mathbb{R}^K : \sum_{k=1}^{K} \pi_k^* = 1, \pi_k^* \geq 0, k = 1, \ldots, K \right\}.$$

The $\delta_{\zeta_k^*}^*$ are an i.i.d. sample of a nonatomic distribution P_0 on Ξ^* (i.e. $P_0(\{\zeta_k^*\}) = 0 \ \forall \zeta_k^* \in \Xi^*$), independent of $\pi_k^*, k = 1, \ldots, K$, and K.

An important subclass of Ongaro-Cattaneo priors are stick-breaking priors. Stick-breaking priors have the Ongaro-Cattaneo form (3.7), but assume that K is fixed (with the possibility of $K = \infty$) and that the weights are sampled according to

$$\pi_1^* = V_1 \quad \text{and} \quad \pi_k^* = V_k \prod_{k^*=1}^{k-1} (1 - V_{k^*}), \tag{3.8}$$

where V_k are independent $\text{Beta}(a_k, b_k)$ random variables, $k = 1, \ldots, K$, and $a_k, b_k > 0$. In general, it is not guaranteed that $\sum_{k=1}^{K} \pi_k^* = 1$. However, it can be shown that assuming $K = \infty$ and $\sum_{k=1}^{\infty} \log(1 + a_k/b_k) = \infty$ ensures that $\sum_{k=1}^{K} \pi_k^* = 1$. If $K < \infty$, it is sufficient to set $V_K = 1$, leading to a generalized Dirichlet distribution for the weights π_k^* (Ishwaran and James, 2001). In Table

3.1, important stick-breaking priors are listed with their corresponding choices for a_k and b_k. All of the priors that are explicitly mentioned in this section may be sampled by a stick-breaking construction.

The term stick-breaking construction is owed to the fact that the sampling may be seen as a process in which pieces are consecutively broken off a stick of length 1, see Figure 3.1. The V_k correspond to the proportion of the remaining stick that is broken off. Wong (1998) discusses sampling from the generalized Dirichlet distribution and the Dirichlet distribution in particular via a stick-breaking construction, independently of their employment in mixture models. Ishwaran and James (2001) describe the sampling of stick-breaking priors in a general framework.

To date, the Dirichlet distribution remains a very frequently used mixture prior in finite mixture models due to its simplicity and its conjugacy properties. Unless specific reasons suggest otherwise, mostly an exchangeable symmetric Dirichlet prior with $\alpha_k \equiv \alpha_0,\ k = 1, \ldots, K$, is chosen that does not favor any particular class. In this case, α_0 is referred to as concentration parameter (the same term and notation is used in the context of the Dirichlet process). A widely used prior is the uniform Dirichlet prior with $\alpha_0 = 1$. However, Ishwaran and Zarepour (2002a) discuss in the context of mixtures of normal distributions that there exist consistency problems with this prior. In particular, they show that this prior is inconsistent when $n/K \to 0$, i.e. when K gets too large in comparison to n. To solve this issue, they recommend to re-parameterize the symmetric Dirichlet prior such that $\alpha_0 = \alpha^*/K$ for an $\alpha^* > 0$ and show that this prior is strongly consistent for the density and weakly consistent for the unknown mixture distribution. Moreover, Ishwaran and Zarepour (2002b) show that it approximates the Dirichlet process for increasing K and that for a fixed value of $\boldsymbol{\zeta}^* = (\zeta_1^*, \ldots, \zeta_K^*)'$ it is already a Dirichlet process. Note that in their considerations, they in general assume that the components of the mixture prior correspond to identical distributions. Green and Richardson (2001) discuss more generally conditions under which the Dirichlet process model is the

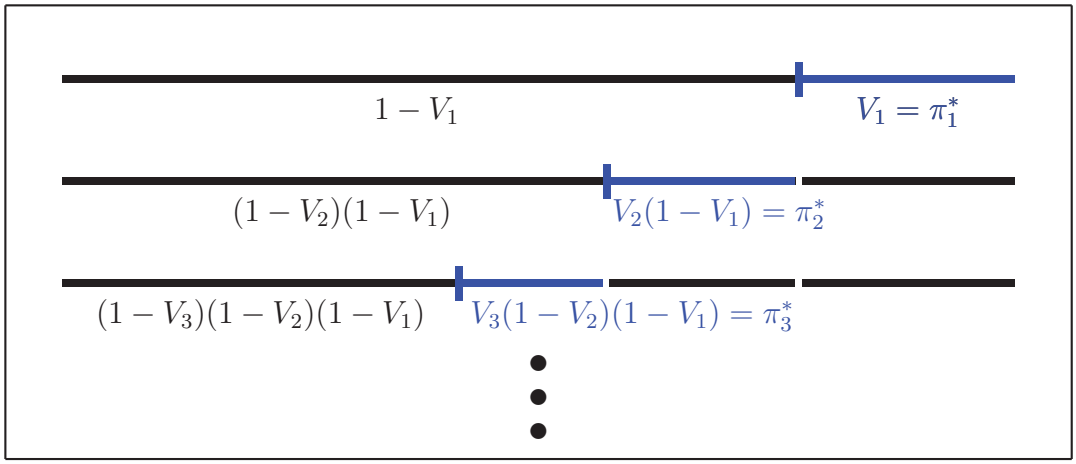

Figure 3.1: Illustration of sampling via a stick-breaking construction.

limiting case of the symmetric Dirichlet prior. The prior resulting from setting $\alpha_k = \alpha^*/K$ for an $\alpha^* > 0$ is referred to with different names, such as Dirichlet-multinomial process (Muliere and Secchi, 1995, stressing its relations to the Dirichlet process), m-spike model (Liu, 1996, in the context of importance sampling), Dirichlet/multinomial allocation model (Green and Richardson, 2001) or finite dimensional Dirichlet prior (Ishwaran and Zarepour, 2002a,b). The latter term is used throughout this thesis to underline its finite-dimensional nature.

The prior clustering behavior of the mixture model depends essentially on the parameter (or the parameters) of the prior distribution chosen for the mixture weights, e.g., on the concentration parameter α_0 in case of the symmetric Dirichlet distribution. When the number of components is chosen by the model and overfitting is present, the concentration parameter α_0 may be very influential. For $\alpha_0 < 1$, e.g., probability mass will concentrate on few components and empty components will a priori be likely, while for $\alpha_0 > 1$ the latter will not be the case (Frühwirth-Schnatter, 2011). Ishwaran and Zarepour (2002a) suggest that $\alpha_0 = 1/K$ is a choice that often works well. However, α_0 might also be estimated as part of the model, avoiding a user-defined choice of the concentration parameter. In case of both the symmetric Dirichlet distribution and the Dirichlet process, the gamma distribution is a popular prior for the concentration parameter. Escobar and West (1995), e.g., employ it in case of

a Dirichlet process and show how the concentration parameter can be sampled within a Gibbs sampler. Ishwaran and Zarepour (2000) advocate the use of a gamma prior also in the context of a symmetric Dirichlet distribution, citing its flexibility. Further employed priors are, e.g., an inverted Beta distribution (Griffin and Steel, 2006) or an inverse gamma distribution (Medvedovic *et al.*, 2004).

In this thesis, models with a fixed small number of components that implicitly control for overfitting are employed based on the research questions and the considered data. The influence of α_0 is, thus, expected to be small, both with regard to its value and to its parametrization (e.g., the one advocated by Ishwaran and Zarepour (2002a) or the uniform Dirichlet prior) .

The prior assigned to $\boldsymbol{\theta}$ depends on the specific form of the $p(y|\theta_k)$. In the literature, the discussion of mixture models is mostly limited to the case in which all mixture components correspond to normal distributions. This is owed to the fact that any density can be approximated by a mixture of normal densities with an arbitrary precision (Ferguson, 1983). Mixture components with other densities are rarely considered. Exceptions can be found in, e.g., Zhang *et al.* (2012), who use a mixture of Beta distributions to model the percentage of methylated alleles, or Svensén and Bishop (2004), who employ Student's t distributions due to its greater robustness. Another example for non-Gaussian mixtures is work by Kottas and Sansó (2007), who employ a mixture of bivariate Beta distributions to account for observations to stem from a limited region of $\mathbb{R}^2$. Densities corresponding to distinct parametric families for different k are usually not considered, but are a focus in this thesis. When no prior information exists, noninformative prior specifications are preferred. Since improper priors such as a uniform distribution with an unbounded support generally lead to an improper posterior for $\boldsymbol{\theta}$ (Roeder and Wasserman, 1997), vague priors are chosen in such cases. In most applications in this thesis, however, prior information not only exists, but is even necessary to a moderate degree for performing the desired inference. Generally, a value chosen a priori for a model parameter θ is

denoted by θ_0 in this thesis. If the parameter already possesses a subscript s, it is denoted by $\theta_{s,0}$.

When considering hyperpriors, conjugacy is often preferred for its computational benefits.

3.1.3 Identifiability and Label Switching

Mixture models in general have the intrinsic property of not being identifiable, i.e. there exist parameter values $(\boldsymbol{\theta}, \boldsymbol{\pi}^*)' \neq (\tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\pi}}^*)'$ so that

$$p(y|\boldsymbol{\theta}, \boldsymbol{\pi}^*) = p(y|\tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\pi}}^*) \text{ for almost all } y \in \mathcal{Y}.$$

The reason for this may be owed to invariance w.r.t. one or more of the three following circumstances: relabeling of the components of the mixture distribution, overfitting, or generic properties of certain classes of mixture distributions.

Permuting the component labels does not change $p(y|\boldsymbol{\theta}, \boldsymbol{\pi}^*)$, since it only changes the order of summation in (3.1). This causes the likelihood (3.3) to have $K!$ identical modes, which propagates to the posterior distribution $p(\boldsymbol{\theta}, \boldsymbol{\pi}^*|y)$ if the priors $p(\theta_k)$ and $p(\pi_k^*)$ are exchangeable, i.e. if they are identical for all k. Multiple identical modes in the posterior distribution lead to problems in the Bayesian estimation of any component-specific parameters such as θ_k, π_k^* or T. However, while the priors $p(\theta_k)$ and $p(\pi_k^*)$ are commonly identical for all k, in this thesis $p(\theta_k)$ differing between distinct values of k are explicitly considered (see Chapter 5), including not only different prior values, but also different distributions. In fact, for no model proposed in this thesis, the priors $p(\theta_k)$ and $p(\pi_k^*)$ are identical for all k. For some k, the priors may be identical, but in these cases the corresponding components jointly represent one group or class and are not distinguished regarding interpretation, making an identification of single components irrelevant.

Nonidentifiability may also be caused by overfitting, i.e. when the number of components in the model exceeds the true number of components, or, if it

is not reasonable to speak of a true number, when it exceeds the number of components that are 'necessary' in a sense depending on the specific modeling task. In particular, nonidentifiability arises when a component is added to a mixture model with K components, and either this component is empty or it is equal to one of the other components, i.e. formally, $\tilde{\pi}^*_{K+1} = 0$ or $\theta_{K+1} = \theta_k$ and $\tilde{\pi}^*_{K+1} + \tilde{\pi}^*_k = \pi^*_k$ for a $k \in \{1, \ldots, K\}$. If desired, e.g., when the number of components in a mixture model is of interest and should be investigated, one may try to avoid the occurrence of $\tilde{\pi}^*_{K+1} = 0$ by choosing $\alpha_0 > 1$ as Dirichlet concentration parameter (Frühwirth-Schnatter, 2011; see also Section 3.1.2). However, this type of nonidentifiability is not relevant for the work presented in this thesis, as only models with a fixed number of components are considered.

The last type of nonidentifiability, caused by specific properties of employed distributions, arises only for densities $p(y|\theta_k)$ from certain parametric families. As an example, consider the random variables

$$\begin{aligned} Y_1 &= \frac{1}{2}U[-2,1] + \frac{1}{2}U[-1,2] \text{ and} \\ Y_2 &= \frac{1}{3}U[-1,1] + \frac{2}{3}U[-2,2], \end{aligned}$$

where U denotes the uniform distribution (Everitt and Hand, 1981). Both random variables have the same distribution, resulting in nonidentifiability problems. However, for finite mixtures of normal and gamma distributions, which are focused in this thesis, Teicher (1963) proves generic identifiability. Note that the exponential distribution, also considered in this thesis, is a special case of the gamma distribution with the shape parameter set to 1.

The practical relevance of identifiability problems in mixture models is owed to the fact that they may result in label switching: if the likelihood (3.3) has several identical modes propagating to the posterior, an MCMC algorithm sampling from the posterior is bound to visit more than one mode if run long enough, since its stationary distribution corresponds to the posterior. In this case, classi-

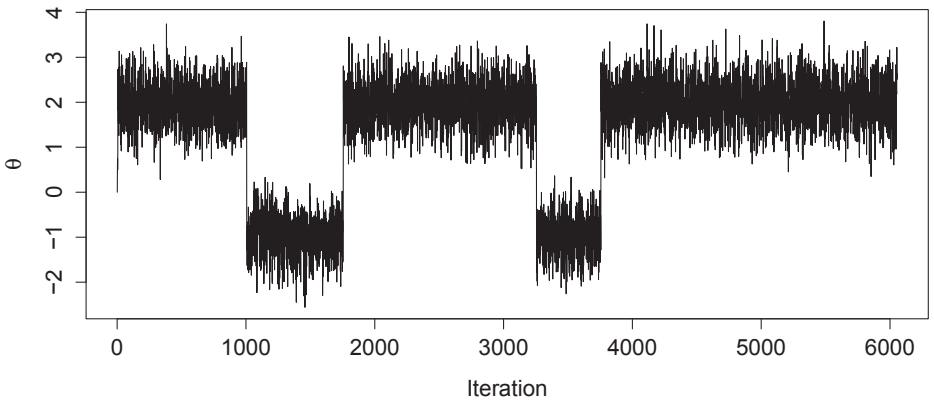

Figure 3.2: Illustration of label switching.

fication of an observation i based on the posterior distribution of the allocation variables T_i becomes difficult, as the component-specific chains and thus the allocation chain will 'jump around' between different values artificially in a way that has nothing to do with real posterior variability (see Figure 3.2). The potential impact of nonidentifiability on the models proposed in this thesis is limited, as explained before, so it is not necessary to worry about label switching as a result of nonidentifiability in their context. Naturally, however, label switching due to real posterior variability cannot be ruled out completely.

3.2 Model Fitting

3.2.1 The Expectation-Maximization (EM) Algorithm

The publication of the work by Dempster *et al.* (1977), formally introducing the EM algorithm, led to an increased employment of finite mixture distributions as a means for modeling heterogenous data. Today, the EM algorithm is still typically the method of choice for fitting mixture models in frequentist statistics and it is often used to fit simple Bayesian models as well. Conceptually, the algorithm estimates parameters for data that is viewed as incomplete.

Specifically, in Section 3.1.1, the data $Y_1, \ldots, Y_n$ can be interpreted as incomplete because their corresponding allocations $T_1, \ldots, T_n$ are unknown. The algorithm solves this incompleteness problem by iteratively calculating the conditional expectation of the log-likelihood for the complete data given $\boldsymbol{y}$, plugging in the current estimates of the unknown parameters in each step.

More precisely, the algorithm iterates between the E-step and the M-step. The E-step adds (and hence updates) the unobserved data $t_1, \ldots, t_n$ to the problem by using Bayes' theorem (3.2) to compute the conditional expectation of $\mathbf{I}(T_i = k)$ for observation i, $i = 1, \ldots, n$, in class k, $k = 1, \ldots, K$, for iteration $\mathbf{t}+1$,

$$\begin{aligned} \hat{T}_{ik}^{(\mathbf{t}+1)} &:= E(\mathbf{I}(T_i = k)|y_i, \boldsymbol{\theta}^{(\mathbf{t})}, \boldsymbol{\pi}^{*(\mathbf{t})}) \\ &= p(T_i = k|y_i, \boldsymbol{\theta}^{(\mathbf{t})}, \boldsymbol{\pi}^{*(\mathbf{t})}) = \frac{\pi_j^{*(\mathbf{t})} p(y|\theta_j^{(\mathbf{t})})}{\sum_{j=1}^{K} \pi_j^{*(\mathbf{t})} p(y|\theta_j^{(\mathbf{t})})}, \end{aligned}$$

given the values $\boldsymbol{\theta}$ and $\boldsymbol{\pi}^*$ from the previous iteration $\mathbf{t}$.

In the M-step, the parameter updates $\theta_k^{(\mathbf{t}+1)}$ and $\pi_k^{*(\mathbf{t}+1)}$ are calculated by maximizing the conditional expectation of the log-likelihood given the data $\boldsymbol{y}$, i.e.

$$\sum_{j=1}^{K} \sum_{i=1}^{n} \hat{T}_{ij}^{(\mathbf{t}+1)} (\log \pi_j^* + \log p(y_i|\theta_j)), \tag{3.9}$$

with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\pi}^*$. If the t_k were observable, the estimates for π_k^* could simply be calculated as

$$\hat{\pi}_k^* = \sum_{i=1}^{n} \frac{\mathbf{I}(T_i = k)}{n}.$$

Replacing the $\mathbf{I}(T_i = k), k = 1, \ldots, K$, with their current conditional expectations $\hat{T}_{ik}^{(\mathbf{t}+1)}$ in the log-likelihood during the E-step leads to

$$\pi_k^{*(\mathbf{t}+1)} = \sum_{i=1}^{n} \frac{\hat{T}_{ik}^{(\mathbf{t}+1)}}{n} \tag{3.10}$$

as updated estimate of π_k^*, where the value of $\boldsymbol{\theta}$ maximizing (3.9) depends on

the form of the $p(y|\theta_k)$. As a nice interpretation, measure (3.10) contains a contribution from each observation i that corresponds to the current estimate for the posterior probability for the allocation of this observation to component k.

The E- and M-steps are repeated in an alternating way until convergence, i.e. until the difference in the values of the maximized log-likelihood from two subsequent iterations becomes arbitrarily small. Dempster *et al.* (1977) show that an EM iteration can never result in a decrease of the likelihood. Therefore, convergence is certain as long as the likelihood has an upper bound.

To apply the EM algorithm, knowledge of the posterior of the unobserved variables given the observations is necessary. This impedes the application of the EM algorithm to complex hierarchical Bayesian models. However, variational Bayes methods constituting an application of the EM algorithm have been developed that attenuate this drawback. These procedures are employed to obtain analytic approximations to the posterior probability of unobserved, missing variables by maximizing a lower bound of the likelihood (see, e.g., Beal and Ghahramani, 2003; Tzikas *et al.*, 2008).

For further details on the EM algorithm, see McLachlan and Krishnan (2008).

3.2.2 Markov Chain Monte Carlo Algorithms

Markov Chain Monte Carlo (MCMC) procedures are the standard method in the Bayesian community to fit not only mixture models, but models in general. Contrary to the EM algorithm, not only point estimates are obtained for the model parameters, but a sample from the posterior distribution of each of these parameters. Integration across the posterior distribution, e.g. to calculate expectations of quantities of interest, can thus be approximated by summarizing across the obtained sample (this approach is called Monte Carlo integration). A Markov chain is a sequence $\{A_1, A_2, A_3, \ldots\}$ of random variables in which A_{t+1}

Algorithm 1 Metropolis-Hastings

If the Markov chain $\boldsymbol{A}$ is in iteration $\mathfrak{t} \geq 0$, carry out the following steps:

1. Draw a candidate B from a proposal distribution $q(. \mid A_{\mathfrak{t}})$, e.g., a (potentially multivariate) normal distribution.
2. Defining φ as the stationary distribution of the Markov chain, accept B with probability

$$\alpha(A_{\mathfrak{t}}, B) = \min\left(1, \frac{\varphi(B)q(A_{\mathfrak{t}} \mid B)}{\varphi(A_{\mathfrak{t}})q(B \mid A_{\mathfrak{t}})}\right), \tag{3.12}$$

   i.e. draw an $U_{MH} \sim U(0,1)$ and set $A_{\mathfrak{t}+1} = B$ if $U_{MH} \leq \alpha(A_{\mathfrak{t}}, B)$. Otherwise set $A_{\mathfrak{t}+1} = A_{\mathfrak{t}}$.
3. Increase the iteration count, i.e. set $\mathfrak{t} = \mathfrak{t} + 1$.

is drawn from the conditional distribution $p(A|A_{\mathfrak{t}})$, i.e. conditional on $A_{\mathfrak{t}}$, $A_{\mathfrak{t}+1}$ is independent of $\{A_1, \ldots, A_{\mathfrak{t}-1}\}$. For any given model parameter, the basic concept of MCMC methods is to construct a Markov chain with a stationary distribution φ that equals the desired posterior distribution

$$p(\boldsymbol{\theta} \mid y) = \frac{p(y \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(y \mid \boldsymbol{\theta})p(\boldsymbol{\theta})\, d\boldsymbol{\theta}} \propto p(y \mid \boldsymbol{\theta})p(\boldsymbol{\theta}). \tag{3.11}$$

This may be achieved by means of the Metropolis-Hastings algorithm (Hastings, 1970), presented in Algorithm 1, or one of its many special cases:

In (3.12), φ is the stationary distribution of the Markov chain independently of the proposal distribution $q(.|.)$, and once the Markov chain reaches the stationary distribution, it will not leave it anymore (see, e.g., Gelman *et al.*, 2013). A good proposal distribution $q(. \mid A_{\mathfrak{t}})$ should be easy to sample from for any $A_{\mathfrak{t}}$, and each accepted jump should go a reasonable distance in the parameter space, since otherwise the chain would move too slowly. Finally, the acceptance ratio $\alpha(A_{\mathfrak{t}}, B)$ should be easy to calculate, and the acceptance rate should be in a reasonable range (Gelman *et al.*, 2013). A very low acceptance rate will cause a long running time of the algorithm, while a very high acceptance rate will typically lead to autocorrelation between the values of the chain, running

contrary to the goal of obtaining an independent sample of draws from the posterior distribution. Thus, in general the goal is to achieve an acceptance rate in the range between 0.1 and 0.9, ideally between 0.3 and 0.7. To ensure this, it may be necessary to tune the standard deviation of the proposal distribution. Autocorrelation nevertheless perceived in a Markov chain after running the algorithm can be reduced by keeping and analyzing only every $\mathfrak{w}$th draw, discarding the others, in a procedure usually referred to as 'thinning'.

Note that although the acceptance probability (3.12) contains the unknown posterior distribution φ, its calculation is possible since the denominators in (3.11) cancel out in the acceptance ratio, leaving only known measures:

$$\frac{\varphi(B)}{\varphi(A_{\mathfrak{t}+1})} = \frac{p(B \mid \boldsymbol{y})}{p(A_{\mathfrak{t}+1} \mid \boldsymbol{y})} = \frac{\frac{p(\boldsymbol{y} \mid B)p(B)}{\int p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})\, d\boldsymbol{\theta}}}{\frac{p(\boldsymbol{y} \mid A_{\mathfrak{t}+1})p(A_{\mathfrak{t}+1})}{\int p(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})\, d\boldsymbol{\theta}}} = \frac{p(\boldsymbol{y} \mid B)p(B)}{p(\boldsymbol{y} \mid A_{\mathfrak{t}+1})p(A_{\mathfrak{t}+1})}.$$

The first $\mathbb{B}$ iterations that pass before the Markov chain reaches the stationary distribution are discarded as burn-in, leaving the remaining sample $(\boldsymbol{\theta}^{\mathfrak{t}}, \varphi^{\mathfrak{t}}), \mathfrak{t} = \mathbb{B}+1, \ldots$ as the basis for inference.

There also exists a componentwise version of the Metropolis-Hastings algorithm, presented in Algorithm 2.

The distributions $\varphi(A_{\cdot\mathfrak{i}}|A_{\cdot-\mathfrak{i}}) = \varphi(A)/(\int \varphi(A)dA_{\cdot\mathfrak{i}})$ are called full conditional distributions. In Algorithm 2, components can be multi-dimensional themselves and may be defined based on the specifics of the model, e.g., correlated components are often grouped into a new component. The grouping may also vary between the iterations.

The componentwise Metropolis-Hastings algorithm owes its importance to one of its special cases in particular, the Gibbs Sampler (Gelfand and Smith, 1990), probably the MCMC procedure most frequently used in Bayesian modeling. Its proposal distribution for the $\mathfrak{i}$th component is given by

$$q_{\mathfrak{i}}(B_{\cdot\mathfrak{i}}|A_{\cdot\mathfrak{i}}, A_{\cdot-\mathfrak{i}}) = \varphi(B_{\cdot\mathfrak{i}}|A_{\cdot-\mathfrak{i}})\,,$$

Algorithm 2 Componentwise Metropolis-Hastings

Partition the Markov chain A into components/subchains $\{A_{\cdot 1}, \ldots, A_{\cdot n}\}$ and define $A_{\cdot -\mathfrak{i}} = \{A_{\cdot 1}, \ldots, A_{\cdot \mathfrak{i}-1}, A_{\cdot \mathfrak{i}+1}, \ldots, A_{\cdot n}\}$. If the Markov chain $A_{\cdot \mathfrak{i}}$ rests in iteration $\mathfrak{t}+1$ and is, thus, still in state $A_{\mathfrak{t}\cdot \mathfrak{i}}$, carry out the following steps:

1. Draw a candidate $B_{\cdot \mathfrak{i}}$ from the proposal distribution $q_{\mathfrak{i}}(B_{\cdot \mathfrak{i}}|A_{\mathfrak{t}\cdot \mathfrak{i}}, A_{\mathfrak{t}\cdot -\mathfrak{i}})$, where

$$A_{\mathfrak{t}\cdot -\mathfrak{i}} = \{A_{\mathfrak{t}+1\cdot 1}, \ldots, A_{\mathfrak{t}+1\cdot \mathfrak{i}-1}, A_{\mathfrak{t}\cdot \mathfrak{i}+1}, \ldots, A_{\mathfrak{t}\cdot n}\}\,.$$

2. Accept $B_{\cdot \mathfrak{i}}$ with probability

$$\alpha(A_{\cdot -\mathfrak{i}}, A_{\cdot \mathfrak{i}}, B_{\cdot \mathfrak{i}}) = \min\left(1, \frac{\varphi(B_{\cdot \mathfrak{i}}|A_{\cdot -\mathfrak{i}}) q_{\mathfrak{i}}(A_{\cdot \mathfrak{i}}|B_{\cdot \mathfrak{i}}, A_{\cdot -\mathfrak{i}})}{\varphi(A_{\cdot \mathfrak{i}}|A_{\cdot -\mathfrak{i}}) q_{\mathfrak{i}}(B_{\cdot \mathfrak{i}}|A_{\cdot \mathfrak{i}}, A_{\cdot -\mathfrak{i}})}\right).$$

3. Set $A_{\mathfrak{t}+1\cdot \mathfrak{i}} = B_{\cdot \mathfrak{i}}$, if $B_{\cdot \mathfrak{i}}$ is accepted. Otherwise, set $A_{\mathfrak{t}+1\cdot \mathfrak{i}} = A_{\mathfrak{t}\cdot \mathfrak{i}}$.

where $\varphi(B_{\cdot \mathfrak{i}}|A_{\cdot -\mathfrak{i}})$ is the full conditional distribution. Thus, the Gibbs Sampler can only be used if the full conditional distributions are known. This is particularly the case when conjugate prior distributions are employed, which is an important reason for the popularity of conjugate models in practice (see Section 3.1.2). The advantage of the Gibbs sampler is that it always accepts the proposed candidate value, since

$$\alpha(A_{\cdot -\mathfrak{i}}, A_{\cdot \mathfrak{i}}, B_{\cdot \mathfrak{i}}) = \min\left(1, \frac{\varphi(B_{\cdot \mathfrak{i}}|A_{\cdot -\mathfrak{i}}) \varphi(A_{\cdot \mathfrak{i}}|A_{\cdot -\mathfrak{i}})}{\varphi(A_{\cdot \mathfrak{i}}|A_{\cdot -\mathfrak{i}}) \varphi(B_{\cdot \mathfrak{i}}|A_{\cdot -\mathfrak{i}})}\right) = 1\,,$$

which typically saves a considerable amount of computation time compared to the general Metropolis-Hastings algorithm.

In case of finite mixture models employing a Dirichlet distribution as prior, a standard implementation of the Gibbs sampler starts from an initial allocation $T^{(0)}$ and then, in iteration $\mathfrak{t}$, alternates between the following steps in which $y_k^{(\mathfrak{t})} = \{y_i : T_i^{\mathfrak{t}} = k\}$ (cf. Algorithm 2):

1. Draw $\boldsymbol{\pi}^{*(\mathfrak{t}+1)} \big| \boldsymbol{T}^{(\mathfrak{t})}$ from the Dirichlet$(\alpha_1 + n_1, \ldots, \alpha_K + n_K)$ distribution, i.e. from (3.6).

2. For $k = 1, \ldots, K$, draw $\theta_k^{(t+1)} \Big| \boldsymbol{y}, \boldsymbol{T}^{(t)}$ from $p(\theta_k | y_k^{(t)})$. If $p(\theta_k | y_k^{(t)})$ is not known, an Metropolis-Hastings step has to be employed here.

3. For $i = 1, \ldots, n$, draw $T_i^{(t+1)} \Big| y_i, \boldsymbol{\theta}^{(t+1)}, \boldsymbol{\pi}^{*(t+1)}$ from the corresponding conditional distribution, i.e. from (3.2).

4. Update chain values and proceed to next iteration.

The EM algorithm and the Gibbs Sampler are structurally similar regarding their sampling from the distributions of one variable conditional on the others. The Gibbs Sampler may thus be interpreted as a stochastic version of the EM algorithm. While both may in principle be used to fit Bayesian mixture models, only the Gibbs sampler may be used to fit complex hierarchical models.

3.2.3 The Number of Components

The number K of components in a finite mixture model has so far been treated as fixed. Both the EM algorithm and standard MCMC algorithms (as discussed in Sections 3.2.1 and 3.2.2) in principle cannot handle a varying dimensionality, i.e. the treatment of K as a random variable. If K is not known, a model selection problem arises. In this situation, it is necessary to choose between models $\mathcal{M}_1, \ldots, \mathcal{M}_{K_{\max}}$, where $\mathcal{M}_K$ denotes a model with K components. There exist a number of model selection criteria intended to guide the choice of K that have the general structure

$$-2 \log p\left(y | \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\pi}}^*, \mathcal{M}_K\right) + C \cdot p_{\mathcal{M}_K}$$

and are minimized to identify the optimal number K. Here, $\hat{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\pi}}^*$ are the estimates given by the model $\mathcal{M}_K$ (i.e., either maximum likelihood estimates or posterior means), $p_{\mathcal{M}_K}$ is a measure for the complexity of $\mathcal{M}_K$, and C is a penalty parameter. Choosing $p_{\mathcal{M}_K}$ as the number of parameters in the model and $C = 2$ or $C = \log(n)$ results in the Akaike information criterion (AIC;

Akaike, 1974) and the Bayesian information criterion (BIC; Schwarz, 1978), respectively. Choosing $C = 2$ and

$$p_{\mathcal{M}_K} = E_{\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\pi}}^*}\left[-2\log p(y|\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\pi}}^*,\mathcal{M}_K)\right] + 2\log p(y|\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\pi}}^*,\mathcal{M}_K) \tag{3.13}$$

leads to the deviance information criterion (DIC; Spiegelhalter *et al.*, 2002), which can be interpreted as a Bayesian analog to the AIC.

Although the generic nonidentifiability of mixture models (as discussed in Section 3.1.3) leads to problems regarding the regularity conditions required for the asymptotic justification of these criteria, they are frequently used to evaluate such models. In the opinion of several authors, both AIC and DIC tend to select overfitted models, i.e. models with K being too large, and corrections or alternatives to tackle this have been proposed for both criteria (see, e.g., Hurvich and Tsai, 1989; Ando, 2007). In practice, the DIC is frequently employed in Bayesian frameworks, while either AIC or BIC are mostly used in frequentist settings. Preferences w.r.t. AIC or BIC differ between authors. While some authors prefer the BIC, as it tends less to overestimate K (Fraley and Raftery, 2002), others favor the AIC for theoretical reasons, e.g., its derivation from principles of information (see Burnham and Anderson, 2002). It has also been proposed to use the BIC to approximate Bayes factors and then employ these factors as criterion (Dasgupta and Raftery, 1998). For Bayesian models fit via MCMC methods, such as the models developed in this thesis, the DIC may be calculated from measures that can be recorded as part of the sampling process, while the calculation of the AIC and BIC requires a separate maximization of the likelihood. In principle this would favor a use of the DIC to evaluate the methods presented in this thesis. However, the aspects underlying the evaluation of the methods developed in this thesis are not fully captured by the measures based on (3.13). Specifically, the method GAMMICS presented in chapter 4 is partly algorithmic, i.e. parts of the estimations done by the method are not represented by the likelihood. The models discussed in chapter 5, on the other hand, are

evaluated based considerably more on classification and interpretability than on the fit. Altogether, thus, the applicability of measures based on (3.13) is limited for these methods.

No matter which of the discussed options is pursued, the model needs to be fitted several times considering different values of K before the resulting models are compared based on the criterion. In a Bayesian context, there exist further options of dealing with an unknown K. If explicit inference on K is required, it is possible to treat K as random and estimate it within the model. In this case, different dimensions of the parameter space have to be considered, and an MCMC algorithm that can deal with $\boldsymbol{\theta}$ and $\boldsymbol{\pi}^*$ of a varying dimension has to be employed. The reversible jump algorithm introduced by Green (1995) fulfills these requirements. Applied to mixture models, it contains moves for adding or removing empty components, as well as for splitting a component in two or fusing two separate components. This ensures that each step of the algorithm is reversible (hence, its name), which is a crucial condition for guaranteeing that Markov chains converge to the desired posterior distribution.

It is also possible to define the number of components via a formal decision-theoretic approach. For instance, one might specify a loss function that reflects the tradeoff between model complexity and the precision in solving the specific inferential problem, e.g. an estimation task (see, e.g., Quintana and Iglesias, 2003; Lau and Green, 2007).

If no explicit inference on K is required, the number of components can be chosen in the sense of an upper bound so that any probable value of K is considerably smaller. The model will then implicitly estimate K by leaving all unnecessary components empty, where the number of non-empty components is influenced by the mixture prior. Such a model (e.g., employing a truncated Dirichlet process or a finite-dimensional Dirichlet prior, as mentioned in Section 3.1.2) has already been applied to model cluster data originating from both biological images (Ji *et al.*, 2009) similar to the data considered in Chapter 4 and omics measurements (Kirk *et al.*, 2012) similar to the data considered in

Chapter 5.

Of course, nonparametric models for infinite mixtures may be preferred from the start, providing more flexibility in terms of fit. Of the potentially infinite number of components in this case, many will typically have weights near zero, however, leading to a limited number of relevant components in practice. In the simplest case, the Dirichlet distribution is then replaced by the Dirichlet process. For an assessment of the influence of different priors on the number of components, see Ishwaran and Zarepour (2000).

In this thesis, the numbers of groups or classes arise naturally from the application at hand. The number of components only needs to exceed the number of groups if one mixture component does not provide sufficient flexibility in terms of fit to represent one group. Furthermore, as mentioned, the focus lies considerably more on classification and interpretability than on the fit. Hence, for the applications considered in this thesis, it appears reasonable to fix K for the modeling tasks.

3.3 Cluster Analysis

Cluster analysis aims to partition observations (or, more general, objects) into groups so that observations within one cluster are as similar to each other as possible, while observations belonging to different clusters are as different as possible. This is of particular interest in exploratory data analysis when only little or even nothing is known about the structure underlying the data. Classical applications include, e.g., the grouping of patients with similar clinical, genetic, or epidemiological patterns to investigate disease causes and to design personalized medicine, or the grouping of clients in a market research data base w.r.t. common consumer behavior, in order to design personalized advertising.

Due to the lack of a training data set with known group allocations from which an allocation rule could be learned, methods for cluster analysis are also termed as unsupervised procedures, as opposed to classification methods that

are called supervised due to training of the allocation rule on a separate data set prior to the classification of new data.

The fact that the allocations to the clusters, the number of clusters, and the characteristics of the clusters are all unknown and have to be learned from the data constitutes a paradox in a way, since it is unclear to some extent what entities something is to be learned about.

The most widely used cluster approaches are likely hierarchical clustering (Johnson, 1967) and partitioning methods such as k-means (Lloyd, 1982; MacQueen, 1967), partitioning around medoids (PAM; Kaufman and Rousseeuw, 1990), or self-organizing maps (SOM; Kohonen, 1995). However, in spite of their popularity, they require input from the analyst in order to render sensible results: the user is obliged to define a cutoff in a dendrogram (hierarchical clustering), pre-define the number of clusters, or infer it by a heuristic (k-means, PAM, SOM). Since the user input usually influences the results heavily, sensitivity analyses are mandatory. Additionally, these approaches do not explicitly consider non-clustered points, although they can produce clusters of size one. Both of the mentioned drawbacks have been tackled by more recent methods. Several cluster modeling approaches take singletons into account, although they are mostly viewed as noise rather than considered as interesting (Banfield and Raftery, 1993; Dasgupta and Raftery, 1998; Hennig and Coretto, 2008). Maitra and Ramler (2009) propose a generalization of the k-means algorithm that explicitly considers scatter points. As for the necessary user input, a number of sophisticated grouping algorithms have been proposed that need less prior information. There are approaches that only require specifying, e.g., a maximum cluster size (Scharl and Leisch, 2006), a minimum cluster size (Manley *et al.*, 2008) or an ε-neighborhood with a minimum point density around core cluster points (Ester *et al.*, 1996; Ankerst *et al.*, 1999). Although such input may be easier to specify, it still requires important prior knowledge on the problem at hand, and the choices made by the user may strongly influence the results. Moreover, sensitivity analyses for the parameter choices are still necessary, but

cannot completely rule out an unwanted bias. Some approaches tackle both drawbacks jointly and also consider the possibility of non-clustered points (such as the one proposed by Scharl and Leisch, 2006). Other approaches are designed for point patterns observed over time and take advantage of point trajectories to find clusters (Manley *et al.*, 2008).

As motivated in Chapter 1, mixture models are a natural approach in the context of cluster analysis. In model-based clustering, observations arise from a mixture of distributions in a given parameter space (Banfield and Raftery, 1993; Fraley and Raftery, 2002). Arbitrary parameter choices are avoided in this framework. Mixture models for clustering may differ structurally in a number of aspects. The most important characteristics in the context of this thesis are: the model fitting technique (basically EM algorithms vs. MCMC methods), the way in which the number of components is chosen (cf. Section 3.2.3), the employed distributions, and the similarity measure used to summarize the relevant information contained in the data.

As discussed in Section 3.1.2, most mixture models in both frequentist and Bayesian approaches exclusively employ Gaussian distributions (for general reference, see McLachlan and Peel, 2000, and Richardson and Green, 1997, respectively). Usually, each cluster is represented by one mixture component, in case of a Bayesian approach at least in each MCMC iteration. However, it frequently happens that at least a few clusters are poorly fitted by a single component in a given application. To overcome the lack of fit in such situations, models involving a mixture of mixtures have been introduced recently. In such approaches, each cluster may be represented by a mixture itself to give the model extra flexibility (see Baudry *et al.*, 2010; Rousseau and Mengersen, 2011, for frequentist and Bayesian approaches, respectively). Bayesian model-based clustering frequently employs nonparametric mixture models, i.e. a potentially infinite number of components, or approximations to such models.

For model-based clustering with non-exchangeable data, more flexible classes of priors have been proposed such as dependent processes (Lijoi *et al.*, 2014).

One might also consider simultaneous clustering of observations and variables, i.e. biclustering (see Wang *et al.*, 2010; Xu *et al.*, 2013), or clustering with implied variable selection that takes into account covariates of varying relevance (Yau and Holmes, 2011). Regarding computation time, Wang and Dunson (2011) have described how to fit a DPM model without using MCMC algorithms (Wang and Dunson, 2011).

Applications of Bayesian model-based clustering are manifold and include, e.g., the clustering of text documents (Blei *et al.*, 2010; Hung *et al.*, 2013), spikes in neuronal signals (Wood *et al.*, 2006), motion patterns (Joseph *et al.*, 2011), or trend curves of cancer incidence and mortality rates (Dass *et al.*, 2014). Methods employing cluster models for the analysis of spatial clustering and omics data are discussed in Chapter 4 and 5, respectively.

In model-based clustering, the cluster partition is generally defined implicitly via the allocation of observations to the mixture components. However, if the cluster partition is of primary interest, it may be an advantage to model the cluster partition directly without specifying a mixture model as an intermediate step. Here, two popular model classes are product partition models (PPMs; Hartigan, 1990; Crowley, 1997; Quintana, 2006) and species sampling models (SSMs; Pitman, 1996; Ishwaran and James, 2003). Both model classes directly assign a probability to the cluster partition. PPMs for this purpose employ cohesion functions that define the similarities of clusters, while SSMs use partition probability functions, i.e. functions of partitions that depend on the partition via the cluster sizes under the assumption of exchangeability. The partition structures induced by PPMs and SSMs are not equivalent in general, but have a considerable intersection. For instance, the marginal prior distribution corresponding to a Dirichlet process can be formulated both in terms of a PPM (Quintana, 2006) and an SSM (Pitman, 1996). Another possibility to specify a mixture model is via a Pólya urn scheme, a generative process that may be illustrated by picking balls of different colors from an urn. The distribution of the mixture weights results as the distribution of the number of balls of each

color in the urn. Every time a ball is drawn, it is replaced by a new ball that is added to the urn. Depending on how this is done, well-known mixture priors such as the Dirichlet distribution or the Dirichlet process can be obtained, or other priors can be constructed.

It is often difficult to average over a sample from the posterior distribution of the cluster partition, or, alternatively, over the posterior distribution of the allocations of each observation. This is in particular the case when the number of clusters is variable or when label switching makes the estimates unstable and inference on cluster specific parameters unreliable (Jasra *et al.*, 2005). This was discussed in Section 3.1.3 in the context of mixture models and, in principle, also applies to PPMs and SSMs (see, e.g., Müller *et al.*, 2011). One possibility to solve this problem is to relabel the cluster labels coherently, as, e.g., proposed by Stephens (2000) for the case of a fixed number of clusters. A solution also applicable to a variable number of components is to choose the clustering that maximizes the posterior density, known as the maximum a posteriori probability (MAP) estimate, a Bayesian analog to the maximum likelihood estimator (Srivastava *et al.*, 2014). Finally, it is possible to average over a posterior sample of similarity matrices instead of over a posterior sample of cluster allocations. This is motivated by the fact that posterior similarity matrices are relatively unsusceptible to label switching (Medvedovic *et al.*, 2004). When research interest focuses on estimating a cluster partition, eventually a partition has to be obtained from the estimated posterior similarity matrix by optimizing a selected criterion. Different estimates have been proposed for this purpose, such as Binder's loss (Binder, 1978), Dahl's criterion (Dahl, 2006), or the posterior expected adjusted Rand index (Fritsch and Ickstadt, 2009).

If only vague ideas exist w.r.t. the clusters' characteristics, regardless of the employed method it is often appropriate to make use of available prior information. This may be done explicitly via corresponding prior distributions in case of a Bayesian analysis. However, it may also be carried out in a more implicit way by defining a distance measure specifically tailored for the clustering

problem at hand (see, e.g., Selinski and Ickstadt, 2008).

In Chapter 4 of this thesis, spatial coordinates of objects are analyzed. In this case, a natural choice for the distance measure is the Euclidean distance. Still, due to the presence of metaclustering and detection errors, informative priors have to be used in the mixture model. In Chapter 5, the information necessary to define distance and similarity between genetic locations is condensed in a specific measure. In both chapters, the analyses can be viewed as clustering. However, they are not clustering in a classical exploratory sense since a fixed number of classes with predefined meanings are considered to which objects are allocated. In this sense, the applications described in this thesis bear more characteristics of classification than of cluster analysis.

Chapter 4

Spatial Modeling of Ras Protein Structures

In this chapter, GAMMICS (GAMma Mixtures for Inference on Cluster Structures) is introduced, a method designed to quantify key parameters of spatial cluster structures formed by Ras proteins. Its performance is compared to several competing approaches. Based on experimental data, it is difficult to evaluate the performance of any of the methods since there is still relatively little well-established knowledge about the cluster behavior of Ras proteins (see Section 2.2). In addition, the Ras patterns obtained by PALM and STORM technologies are subject to unknown detection errors committed by the preprocessing software: erroneous deletions or false detections of proteins may propagate into the subsequent analysis and cause biases in the results. In consequence, an assessment of the methods' performance in a comprehensive simulation study is mandatory. In simulations, the real cluster structure is known and potential strengths or weaknesses of the methods can be systematically investigated w.r.t. different aspects of the data. Thus, in Section 4.1, some theory on point process models is briefly revised. In Section 4.2, a point process model designed specifically for simulating Ras protein patterns is described. For further details on point process theory refer to, e.g., Cressie (1993), Diggle (2003) or Illian

et al. (2008). Sections 4.5 is dedicated to the new GAMMICS method, while in Section 4.4 competing approaches are discussed. In Section 4.7, results obtained by all methods are reported. The contents of most parts of this chapter are the subject of Schäfer *et al.* (2015).

4.1 Point Processes

A spatial point pattern can easily be defined by the locations of its observations $x_1, \ldots, x_n$, assumed to be in $\mathcal{X} \subset \mathbb{R}^\mathfrak{d}$. Alternatively, a spatial point pattern may be defined through a counting measure ϕ on $\mathcal{X}$: for a Borel set $\mathcal{A} \in \mathfrak{Z}$, let $\phi(\mathcal{A})$ be the number of observations in $\mathcal{A}$, where $\mathfrak{Z}$ is the Borel σ algebra of $\mathcal{X}$. Assuming ϕ is locally finite, i.e. $\phi(\mathcal{A}) < \infty$ for all bounded sets $\mathcal{A} \in \mathfrak{Z}$, the knowledge of $\phi(\mathcal{A})$ for all $\mathcal{A} \in \mathfrak{Z}$ renders the same information about the point pattern as the locations of all observations $x_1, \ldots, x_n$ in $\mathcal{X}$.

A spatial point pattern can be viewed as the realization of a spatial point process, i.e. a stochastic model to represent the locations of a countable set of observations $\{x_i\}$. Each of the two mentioned definitions of a point pattern leads to a characterization of a spatial point process, and Moyal (1962) shows that both characterizations of point processes are equivalent. Formally, let $(\Omega, \mathfrak{A}, P)$ be a probability space and let Φ be a collection of locally finite counting measures on $\mathcal{X} \subset \mathbb{R}^\mathfrak{d}$. A spatial point process $\mathcal{N}$ on $\mathcal{X}$ then is a measurable mapping of $(\Omega, \mathfrak{A})$ into $(\Phi, \sigma(\Phi))$ inducing a probability measure $\Pi_\mathcal{N} \equiv P(\mathcal{N} \in Y_\Phi)$ on $(\Phi, \sigma(\Phi))$ for all $Y_\Phi \in \sigma(\Phi)$, where $\sigma(\Phi)$ is the smallest σ algebra generated by sets of the form $\{\phi \in \Phi : \phi(\mathcal{A}) \in \mathbb{N}_0\}$ (Cressie, 1993).

Alternatively, point processes can be defined more intuitively as a model for point locations. For this purpose, $\mathfrak{X} \subset \mathbb{R}^\mathfrak{d}$ is viewed as the set of all point patterns $\boldsymbol{x^n} = \{x_1, \ldots, x_n\}$ with $n = 0, 1, 2, \ldots$, where $\boldsymbol{x^0}$ denotes an empty point pattern. Then a triplet $(\mathfrak{X}, \mathfrak{B}, P^*)$, where $\mathfrak{B}$ is a σ algebra of sets in $\mathfrak{X}$ and P^* a probability distribution on $\mathfrak{B}$, constitutes a model of a stochastic population and is called a spatial point process (Moyal, 1962). A probability

measure can be defined here as well.

If for two point processes $\mathcal{N}_1$ and $\mathcal{N}_2$ it is $\Pi_{\mathcal{N}_1}(Y_\Phi) = \Pi_{\mathcal{N}_2}(Y_\Phi)$ for all $Y_\Phi \in \sigma(\Phi)$, they are said to be identically distributed. A point process is said to be simple if $\phi(x) \in 0, 1$ for all $x \in \mathcal{X}$ and almost all $\phi \in \Phi$. In the following, finiteness and measurability will be assumed for discussed point processes. In addition, $\mathfrak{d} = 2$ will be assumed, based on the Ras protein application focused in this thesis.

4.1.1 Properties of Spatial Point Processes

Frequently, there is only one realization, i.e. a single point pattern to learn about an underlying spatial point process. To make the fit of a model and the estimation of its parameters easier in these situations, often the assumptions of stationarity and, sometimes, isotropy are made. Formally, let $\tau_\varsigma : \mathbb{R}^2 \to \mathbb{R}^2$ be the translation operator $\tau_\varsigma(x) \equiv x - \varsigma$ for each $\varsigma \in \mathbb{R}^2$, let $\phi_\varsigma(\mathcal{A}) \equiv \phi\tau_\varsigma^{-1}(\mathcal{A}) = \phi(\{x : x + \varsigma \in \mathcal{A}\})$ for each $\phi \in \Phi$ and let $N_\varsigma \equiv N\tau_\varsigma^{-1}$. Then, a point process $\mathcal{N}$ is called stationary if $\mathcal{N}$ and $\mathcal{N}_\varsigma$ are identically distributed for all $\varsigma \in \mathbb{R}^2$, i.e. if $\Pi_{\mathcal{N}}(\mathcal{A}) = \Pi_{\mathcal{N}_\varsigma}(\mathcal{A})$ for all $\mathcal{A} \in \mathfrak{Z}$ and for all $\varsigma \in \mathbb{R}^2$. In other words, stationarity means that a process' properties are invariant under translations τ_ς. A point process $\mathcal{N}$ is said to be isotropic if it is invariant under rotations ρ_ϑ about the origin (by an angle ϑ), i.e. when $\mathcal{N}$ and $\mathcal{N}\rho_\vartheta^{-1}$ are identically distributed for all $\vartheta \in [0, 2\pi)$. Note that neither stationarity nor isotropy rule out random heterogeneity.

When analyzing compact regions in $\mathbb{R}^2$, as in this thesis, it is often sufficient to postulate (approximate) stationarity and isotropy within the study regions. However, in the case of Ras protein applications, where complex cluster patterns are expected, stationarity and isotropy can generally not be assumed. Departures from these assumptions may be reduced by choosing relatively small regions of interest located at a sufficient distance away from cell boundaries, at which sharp changes of the point pattern structure are expected to occur.

The first order properties of a spatial point process are described by an intensity function:

Definition 4.1 (**Intensity function**).
The intensity function $\lambda(x)$ *is defined as*

$$\lambda(x) = \lim_{\nu(dx)\to 0} \left\{ \frac{E[\mathcal{N}(dx)]}{\nu(dx)} \right\},$$

where $\mathcal{N}(\mathcal{A})$ *is the number of points in* $\mathcal{A}$*,* dx *is an infinitesimal region which contains* x*,* $E[\mathcal{N}(\mathcal{A})] = \int_\Phi \phi(\mathcal{A})\Pi_\mathcal{N} d\phi$ *and* $\nu(\mathcal{A})$ *is the Lebesgue measure, i.e.* $\nu(\mathcal{A}) = |\mathcal{A}|$.

For a stationary process, it is $\lambda(x) \equiv \lambda$, i.e. the intensity function assumes a constant value. The second-order intensity function is defined by

$$\lambda_2(x_A, x_B) = \lim_{\nu(dx_A)\to 0, \nu(dx_B)\to 0} \left\{ \frac{E[\mathcal{N}(dx_A)\mathcal{N}(dx_B)]}{\nu(dx_A)\nu(dx_B)} \right\}.$$

For a stationary process, $\lambda_2(x_A, x_B)$ can be written as a function λ_2^* of the difference between x_A and x_B, i.e. $\lambda_2(x_A, x_B) = \lambda_2^*(x_A - x_B)$. If, in addition, the process is isotropic, it can be written as a function λ_2^o of the distance between x_A and x_B, i.e. $\lambda_2(x_A, x_B) = \lambda_2^o(\|x_A - x_B\|)$.

Ripley's K-function (Ripley, 1977; Dixon, 2002) and other functions derived from it are related second moment measures. They play a prominent role in the literature on the quantification of Ras protein clusters. The K-function is defined as follows:

Definition 4.2 (**Ripley's K-function**).
For a simple, stationary and isotropic spatial point process with intensity λ*, the K-function is defined as:*

$$\mathcal{K}(r) = \frac{E(\mathcal{N}_0(r))}{\lambda}; \quad r \geq 0 \tag{4.1}$$

where $\mathcal{N}_0(r)$ is the random number of observations lying within a distance of no more than r of any observation.

In the conditions of Definition 4.2, let $\lambda_2^o(r)$ be the second-order intensity. Then, the relation of the second-order intensity and the K-function is

$$\lambda_2^o(r) = \frac{\lambda^2 \Gamma(2)}{2\pi r} \mathcal{K}'(r).$$

(for a general $\mathfrak{d}$, this formula is given in Cressie, 1993). For a given spatial point pattern defined in $R_\mathcal{K} \in \mathbb{R}^2$ containing the point process points $x_1, \ldots, x_n$, $\mathcal{K}(r)$ can be estimated by

$$\hat{\mathcal{K}}(r) = \frac{1}{\hat{\lambda} \cdot n} \sum_{i=1}^{n} \sum_{j \neq i} \mathbf{I}(\|x_i - x_j\| \leq r), \tag{4.2}$$

where $\mathbf{I}$ is the indicator function. The measure can be adjusted for edge effects in several ways (Goreaud and Pélissier, 1999). For instance, in the summation in (4.2) all x_i located at a distance of less than r to the nearest edge of $R_\mathcal{K}$ may be excluded. Alternatively, a weighted sum over i can be used as estimator, calculating the weights for each x_i via a circle around x_i with radius r. Each weight would correspond to the proportion of the area of this circle that lies inside of $R_\mathcal{K}$.

The function $\mathcal{K}$ compares a given point pattern with a homogenous one. Specifically, $\mathcal{K}(r)$ can be interpreted as the area covered by the number of points expected within distance r from an arbitrary point of the given pattern under homogeneity. If there is no structure at all in the observed point pattern, then $\mathcal{K}(r) = \pi r^2$, while $\mathcal{K}(r) > \pi r^2$ indicates a tendency to clustering and $\mathcal{K}(r) < \pi r^2$ a tendency to a regular pattern (see Section 4.1.2)..

To provide a user-friendly interpretation, the K-function is often normalized, e.g. to $\mathcal{L}(r) = \sqrt{\frac{\mathcal{K}(r)}{\pi}}$ or to $\mathcal{H}(r) = \mathcal{L}(r) - r$. The key value for comparison in the assessment of clustering is then no longer $2\pi r^2$, but r or 0, respectively. For the

K-function, information across a range of radii is accumulated. To summarize the information contained in the K-function without accumulative effects, one may consider the pair correlation function.

Definition 4.3 (**Pair correlation function**)**.**
In the conditions of Definition 4.2, the pair correlation function is defined as:

$$g(r) = \frac{d\mathcal{K}(r)}{dr}/(2\pi r); \quad r \geq 0.$$

Both the K-function and the pair correlation function may be used to assess whether a point pattern stems from a specific point process model or not. The functions may also be used to fit such a model to a point pattern, using the similarity between theoretical and empirical values as a goodness-of-fit criterion. This is called the method of minimum contrast, see, e.g., Diggle (2003) or Stoyan and Stoyan (1994). However, as demonstrated by Baddeley and Silverman (1984), there are point processes that share the same second order characteristics but nevertheless differ notably in other respects.

4.1.2 Spatial Point Process Models

One of the most important point processes in spatial statistics is the Poisson process, which is the basis for many point process models. It is defined as follows:

Definition 4.4 (**Poisson process**)**.**
$\mathcal{N}$ is an (inhomogenous) Poisson process if:

1. *For any $\mathcal{A} \in \mathfrak{Z}$, $P(\mathcal{N}(\mathcal{A}) \in \{0, 1, \ldots\}) = 1$, and for any collection of disjoint sets $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k \in \mathfrak{Z}$, the random variables $\mathcal{N}(\mathcal{A}_1), \mathcal{N}(\mathcal{A}_2), \ldots,$ $\mathcal{N}(\mathcal{A}_k)$ are independent.*

2. For all $x \in \mathcal{X}$*,*

$$\begin{aligned} P(\mathcal{N}(dx) = 0) &= 1 - E[\mathcal{N}(dx)] + o(E[\mathcal{N}(dx)]), \\ P(\mathcal{N}(dx) = 1) &= E[\mathcal{N}(dx)] + o(E[\mathcal{N}(dx)]), \\ P(\mathcal{N}(dx) > 1) &= o(E[\mathcal{N}(dx)]), \end{aligned}$$

provided that $E[\mathcal{N}(\mathcal{S})]$ *exists for any compact set* $\mathcal{S} \in \mathfrak{Z}$*.*

It can be shown that under the circumstances described in Definition 4.4, $\mathcal{N}(\mathcal{A})$ has a Poisson distribution with mean $E[\mathcal{N}(\mathcal{A})]$ for all $\mathcal{A} \in \mathfrak{Z}$ (see, e.g. Rogers, 1974). This fact, besides the Poisson distribution's status as a popular choice for modeling count data, motivates why Poisson processes are so important for the modeling of point processes.

In the special case of $E[\mathcal{N}(\mathcal{A})] = \lambda \cdot \nu(\mathcal{A}) = \lambda|\mathcal{A}|$ for a $\lambda > 0$ for all $\mathcal{A} \in \mathfrak{X}$, where $\nu(\mathcal{A}) = |\mathcal{A}|$ is the Lebesgue measure, the Poisson process is homogenous, describing the absence of any structure in spatial point data. This important case is also known as complete spatial randomness.

There are two basic deviations from complete spatial randomness: regularity and clustering. While points of a regular pattern tend to display mutual inhibition, i.e. there is a minimum mutual distance, clustered points tend to form agglomerations.

There are several possible classes of models for simulating point patterns with clusters. The perhaps most obvious one is the class of independent cluster processes, constructed by means of a parent and a daughter process: cluster centers arise independently from the parent process, while the points in each cluster are conditionally independent, arising from a daughter process. A popular subclass are Neyman-Scott cluster processes (Neyman and Scott, 1958), defined via a Poisson parent process and a Poisson daughter process. Different distributions are considered to model the point locations within each cluster, e.g., a uniform or normal distribution.

Independent cluster processes imply the possibility of mutually overlapping points, an undesirable characteristic when modeling objects of a certain physical size. Thus, one may consider a notion of regularity in the model. The easiest way of doing this is by simple inhibition point processes, e.g., so called hard core models which establish overlap-free circles around simulated points. One prominent example are the models of Matérn (Matérn, 1960) which delete points violating the non-overlap rule.

A different approach is offered by Markov point processes, a more complex class of mainly inhibitive point processes formulating specific assumptions about potential interactions between points (Ripley and Kelly, 1977). Among the earliest proposed subclasses are pairwise interaction processes (Ripley, 1977; Strauss, 1975) which describe dependence between points based on the notion of R-nearness of point pairs: two points whose distance is less than R are considered to be in interaction. More sophisticated alternatives have since been proposed such as area interaction processes (Baddeley and van Lieshout, 1995), constituting a generalization to interactions of potentially infinite order. Connected component Markov point processes (Baddeley and Møller, 1989) also consider points connected by a long chain of points as being in interaction. Grabarnik and Särkka (2001) introduce the class of interacting neighbor point processes in which points are regarded as interacting when they are either sufficiently close or connected by a single third point.

Ras proteins are physical objects of a certain size and thus cannot overlap, i.e. the locations of two Ras proteins cannot be arbitrarily close. This would in principle be an argument to employ an inhibitive process for their simulation. However, neither the existing simple inhibition processes of Matérn and extensions nor the different Markov point processes provide the possibility of combining local inhibition with the more important feature of clustering. Furthermore, in case of the Ras proteins the inhibition loses importance because the proteins can only be detected indirectly by PALM/STORM due to their size, involving a detection error that is several times higher than the radius of

parent points η	offspring ζ	name of process
general	general	independent cluster process
Poisson	general	Poisson cluster process
general	Poisson	Cox cluster process
Poisson	Poisson	Neyman-Scott process
Poisson (homogen.)	Poisson (uniform in disc)	Matérn cluster process
Poisson (homogen.)	Poisson (Gaussian)	Modified Thomas process

Table 4.1: Standard nomenclature for independent cluster processes.
Source: van Lieshout and Baddeley (2002).

the proteins. Any gain in representing local inhibition between Ras proteins will, therefore, be masked by the detection error, casting doubt on the added value of the more complex model for this application.

For these reasons, in the following the focus is put on spatial cluster processes. In particular, a new subclass, the double Matérn cluster process, is described as an appropriate framework to simulate Ras protein patterns.

4.1.3 Spatial Cluster Processes

To formally define cluster processes, let $\boldsymbol{\eta} = \{\eta_1, \ldots, \eta_{n_\eta}\}$ be a first generation point process on $\mathbb{R}^2$ and let $\boldsymbol{\zeta} = \{\zeta_1, \ldots, \zeta_{n_\zeta}\}$ be second generation points grouped in conditionally independent clusters $\mathcal{C}_1, \ldots, \mathcal{C}_{n_\mathcal{C}}$ centered at the $\eta_j, j = 1, \ldots, n_\eta$, where $|\mathcal{C}_j| \geq 1$. The union $\mathcal{U} = \cup_j \mathcal{C}_j$ is called an independent cluster process. A corresponding point pattern consists of a realization of $\mathcal{U} \cap \Psi$ in a non-empty compact window $\Psi \subset \mathbb{R}^2$. Note that some η may be located outside of Ψ and that properties of the clusters, such as the intensity within clusters or spread of the clusters, may vary. For each η there is a point process with distribution $G_\mathcal{C}$ and density $g_\mathcal{C}(\cdot)$, representing the spread of the cluster corresponding to η. In Table 4.1, the nomenclature for special cases of independent cluster process models is listed.

In this thesis, the focus lies on Neyman-Scott processes, a special case of

Cox cluster processes (Cox, 1955) in which both the fist and second generation points follow Poisson processes. A standard Neyman-Scott process is defined as follows:

Definition 4.5 (**Neyman-Scott cluster process**).
A Neyman-Scott process fulfills the following conditions:

NS1 *There are first generation points $\eta_1, \ldots, \eta_{n_\eta}$ that form a Poisson process.*

NS2 *Each first generation point produces a cluster $\mathcal{C}_j$ consisting of a random number S_c of second generation points, where $S_c \sim Poi(\mu_c)$ i.i.d.*

NS3 *The positions of the second generation points $\zeta_1, \ldots, \zeta_{n_\zeta}$ relative to their respective parent points are i.i.d. distributed, given their number, according to the density function $g_{\mathcal{C}}(\cdot) = \xi(\cdot|\eta, r)/\Xi(\eta)$ on $\mathbb{R}^2$, with an intensity function $\xi(\cdot|\eta, r) : \mathbb{R}^2 \to [0, \infty)$, where $\Xi(\eta) = \int \xi(t|\eta, r)dt = \mu_c$ and r is the cluster radius.*

By the superposition property of Poisson processes, conditional on $\boldsymbol{\eta}$, the combined clusters form a Poisson point process on $\mathbb{R}^2$ with intensity function

$$\lambda(\cdot|\boldsymbol{\eta}, \boldsymbol{r}) = \sum_{j=1}^{n_\eta} \xi(\cdot|\eta_j, r)$$

with the convention that $\xi(\mathfrak{u}|\cdot, r) = 0$ if $\mathfrak{u} \notin \Psi$. Often, the intensity function for one cluster $\xi(\mathfrak{u}|\eta, r)$ essentially only depends on the distance $\text{dist}(\eta, \mathfrak{u})$ between η and $\mathfrak{u}$. Here, it is written additionally in dependence of the cluster radius $r > 0$, which is one of the parameters of interest in the Ras protein analyses. Generally, it is not obvious how r is defined. One interesting definition for $\xi(\mathfrak{u}|\eta, r)$ is

$$\xi(\mathfrak{u}|\eta, r) = \frac{\gamma}{r}\sqrt{\frac{2}{\pi}} \cdot e^{-2 \cdot \text{dist}(\eta, \mathfrak{u})^2/r^2}, \tag{4.3}$$

for a $\gamma > 0$, i.e. the daughters follow an isotropic Gaussian distribution with center η. This is known as the modified Thomas process (Thomas, 1949). Here,

σ is defined as $r/2$ to write $\xi(\mathfrak{u}|\eta, r)$ as a function of the radius r, departing from the conventional parametrization of a Gaussian distribution. Another important example is

$$\xi(\mathfrak{u}|\eta, r) = \begin{cases} \gamma & \text{if } \operatorname{dist}(\eta, \mathfrak{u}) \leq r \\ 0 & \text{otherwise,} \end{cases} \tag{4.4}$$

which is known as the Matérn cluster process (Matérn, 1986). More generally than in these examples, the point spread may depend on η.

Two properties of Neyman-Scott processes follow from the fact that the clusters arise from Poisson processes (i.e. they are Cox cluster processes). First, conditional on $\boldsymbol{\eta}$ and n_ζ, the points are drawn from a finite mixture distribution with n_η mixture distributions determined by the $\eta_j, j = 1, \ldots, n_\eta$, and weights

$$p_j = \frac{\Xi(\eta_j)}{\sum_{m=1}^{n_\eta} \Xi(\eta_m)}, j = 1, \ldots, n_\eta,$$

see van Lieshout and Baddeley (2002). If the intensity function ξ is translation invariant in the sense that $\xi(\mathfrak{u}+\varsigma|\eta, r) = \xi(\mathfrak{u}|0, r)$ for all $\varsigma \in \mathbb{R}^2$, the weights are identical for all parents. While common in the spatial context, this restriction is rather unnatural in the general context of finite mixture models.

Second, clusters have a positive probability of being empty and of containing only one point. If in a specific application this is an undesirable feature, steps may be taken to avoid such parents. One easy option would be, e.g., to consider only such parents having at least one (or more) daughters.

In Section 4.2, a point process model more general than the Neyman-Scott process is presented, reflecting characteristics of point patterns arising in the analysis of Ras proteins.

For a more profound discussion of spatial cluster processes see also, e.g., Lawson and Denison (2002) or Daley and Vere-Jones (2008).

4.2 The double Matérn Cluster Process

The Ras protein structures motivating this chapter have properties not reflected by established Neyman-Scott processes. First, only a portion of the proteins does actually cluster, some of them appear as singletons. Second, Ras clusters as well as singletons may (jointly) form considerably bigger meta structures similar to clusters, which will be called 'metaclusters' in the following. For these reasons, a more complex cluster process model is needed for simulation purposes that allows for both singletons and such metaclusters.

In the context of spatial point patterns in ecology, such as the location of tree species, Watson *et al.* (2007) and Wiegand *et al.* (2007) discuss and apply a nested form of the modified Thomas process, termed 'double-cluster' process. Diggle (2003) considers the more general concept of 'multi-generation' processes with several potential levels of clustering in which the offspring points become the parent points of the respective next generation. He does not recommend any specific distributional assumptions for the point spread within clusters. Wiegand *et al.* (2009) propose to combine the 'double-cluster' process of Watson *et al.* (2007) and Wiegand *et al.* (2007) with a superposed homogenous Poisson process for singletons ('double-cluster–random superposition'). They fit different point process models with the method of minimum contrast, employing the pair correlation and L-functions as criterion. As cluster-generating process, they choose the modified Thomas process because it results in a mathematically simple form of the pair correlation function. They also argue that a point density declining with increasing distance from the cluster center is biologically meaningful in their frugivore-dispersed tree species examples. In the Ras cluster application, however, the latter argument is difficult to establish, particularly due to the measurement error inherent to the analyzed point patterns. Also, when the point spread within clusters follows a Gaussian distributions, there is a small but positive probability of points lying too far from the cluster center to be recognized correctly as part of the cluster. For these reasons, in this the-

sis the Matérn cluster process is employed to model the points spread within clusters instead, providing more flexibility in the point distribution within clusters. The Matérn cluster process has already been employed for simulations of Ras proteins in previous works within the Ras protein research project in Katja Ickstadt's working group (Müller-Heine, 2009; Hebestreit, 2009).

Fitting models that involve nested clustering is difficult due to confounding between clustering and metaclustering. Diggle (2003) acknowledges that they tend to be mathematically intractable. Thus, to fit their 'double-cluster process' via the pair correlation function, Wiegand *et al.* (2009) employ a two-step fitting process involving a comparison of plots in order to manually assess the rough order of magnitude of clustering parameters on the two clustering levels. In one step, this manual procedure results in the restriction of parameter values to a certain scale. Regarding the incorporation of a component for singleton points, a Poisson cluster process and its combination with a random superposition process in general are not distinguishable based on their second order properties. However, they are distinguishable based on their specific nearest neighbor properties (Diggle, 2003). Thus, to fit their double-cluster–random superposition model, Wiegand *et al.* (2009) aim to find a cutoff distance separating isolated and clustered points based on the distribution of nearest neighbor distances, comparing the fit based on several possible cutoffs. In this thesis, a similar idea based on distances to second nearest neighbors will be employed to distinguish clustered points and singletons. However, it will be implemented in a more comprehensive, model-based way that moreover avoids any manual assessment of clustering parameters on the two clustering levels based on plots. In addition, no distributional assumptions for the point spread within clusters are made when fitting a model (see methodology described in Section 4.5).

Assuming the 'double-cluster' process of Wiegand *et al.* (2007) as nested clustering mechanism has two disadvantages: first, there is no possibility of clusters lying outside of metacluster structures. Second, the probability of metaclusters of size one cannot be regulated independently of the metacluster size

due to sampling the cluster and metacluster sizes from a Poisson distribution. In this section, an alternative cluster process definition is proposed for simulation purposes. While not considering singletons as an independent component, it permits to freely choose both the proportions of clustered and singleton points and the cluster size at each level of the nested clustering. This leads to a greater flexibility regarding the simulated point patterns, permitting clusters lying outside of metaclusters. Specifically, a cluster process is proposed that may be viewed as a generalized version of the Matérn cluster process. Points have a three-generation hierarchical structure. The process is termed double Matérn cluster process and is formally defined as follows:

Definition 4.6 (**Double Matérn cluster process**)**.**
A double Matérn cluster process with support on a compact region $\Psi \subset \mathbb{R}^2$ *and parameters* $n_{total} \in \mathbb{N}$, $p_c \in [0,1]$, $p_{mc} \in [0,1]$, $\mu_c \in \mathbb{R}^+$, $\mu_{mc} \in \mathbb{R}^+$, $r_c \in \mathbb{R}^+$ *and* $r_{mc} \in \mathbb{R}^+$ *can be defined via the following postulates:*

DM1 *There are first generation points* $\boldsymbol{\eta} = \{\eta_1, \ldots, \eta_{n_\eta}\}$ *arising from a Poisson process.*

DM2 *With probability* $\tilde{p}_{mc}$*, each first generation point produces a metacluster consisting of a random number* S_{mc} *of offspring points (second generation points), where* $S_{mc} \sim Poi(\mu_{mc})$ *i.i.d.*

DM3 *The positions of the second generation points* $\boldsymbol{\zeta} = \{\zeta_1, \ldots, \zeta_{n_\zeta}\}$ *relative to their respective parents are i.i.d. distributed, given their number, according to the density function* $g_{\mathcal{MC}}(\cdot|R_{mc}) = \xi(\cdot|\eta, R_{mc})/\Xi(\eta)$ *on* $\mathbb{R}^2$*, where* $\xi(\cdot|\eta, R_{mc})$ *is defined by (4.4) with* $\Xi(\eta) = \int \xi(x|\eta, R_{mc})dx = \mu_{mc}$ *and* $\eta \in \boldsymbol{\eta}$*.* R_{mc} *is assumed to be a random variable ensuring that the expected metacluster radius is* r_{mc}*.*

DM4 *With probability* $\tilde{p}_c$*, the existent points (both first generation points not replaced by a metacluster, as well as the second generation points) pro-*

duce a cluster consisting of a random number S_c of offspring points (third generation points), where $S_c \sim Poi(\mu_c)$ i.i.d.

DM5 *The positions of the third generation points relative to their respective parents are i.i.d. distributed, given their number, according to the density function $g_{\mathcal{C}}(\cdot, R_c) = \xi(\cdot|\tilde{x}, R_c)/\Xi(\tilde{x})$ on $\mathbb{R}^2$, where $\xi(\cdot|\tilde{x}, R_c)$ is defined by (4.4) with $\Xi(\tilde{x}) = \int \xi(x|\tilde{x}, R_c)dx = \mu_c$ and $\tilde{x} \in \boldsymbol{\eta} \cup \boldsymbol{\zeta}$. R_c is assumed to be a random variable ensuring that the expected metacluster radius is r_c.*

The quantities n_η, $\tilde{p}_{mc}$ and $\tilde{p}_c$ used in Definition 4.6 are defined and justified in the following theorem:

Theorem 4.1 (**Key quantities for sampling from double Matérn cluster process**)**.**
When a proportion of

$$\tilde{p}_{mc} = \begin{cases} \frac{1}{\frac{\mu_{mc}}{p_{mc}} - \mu_{mc} + 1} & \text{if } p_{mc} \neq 0, \\ 0 & \text{if } p_{mc} = 0, \end{cases} \tag{4.5}$$

of first generation points is replaced by metaclusters containing a random number S_{mc} of second generation points, where $S_{mc} \sim Poi(\mu_{mc})$, and a proportion of

$$\tilde{p}_c = \begin{cases} \frac{1}{\frac{\mu_c}{p_c} - \mu_c + 1} & \text{if } p_c \neq 0, \\ 0 & \text{if } p_c = 0, \end{cases} \tag{4.6}$$

of all the existent points (first and second generation points) is replaced by a cluster containing S_c third generation points, where $S_c \sim Poi(\mu_c)$, then the overall proportion of clustered points will be approximately p_c. Starting with

$$n_\eta = \frac{n_{total}}{[\tilde{p}_c(\mu_c - 1) + 1] \cdot [\tilde{p}_{mc}(\mu_{mc} - 1) + 1] \cdot \left[1 - \frac{1-p_c}{p_c} \cdot \tilde{p}_c^2 - \frac{1-p_{mc}}{p_{mc}} \cdot {\tilde{p}_{mc}}^2\right]}$$

first generation points, one obtains a point pattern consisting of a total of approximately n_{total} points.

Algorithm 3 Simulating from the double Matérn cluster process

1. A number of n_η first generation points is sampled from a homogenous spatial Poisson process in Ψ,
2. A proportion $\tilde{p}_{mc}$ of first generation points is randomly replaced by metaclusters containing a random number S_{mc} of second generation points distributed uniformly within a radius of R_{mc}, where $S_{mc} \sim \text{Poi}(\mu_{mc})$, and $R_{mc} \sim \text{Gamma}(r_{mc}, 1)$.
3. Of all points existent at this stage, a proportion of $\tilde{p}_c$ is replaced by clusters containing S_c third generation points distributed uniformly within a radius of R_c, where $S_c \sim \text{Poi}(\mu_c)$, and $R_c \sim \text{Gamma}(r_c, 1)$.

To serve the needs of the application to Ras proteins, one may add:

4. Every remaining point $x \in \mathbb{R}^2$ is replaced by a point randomly drawn from $N(x, \sigma_d^2 \text{I})$, where $\sigma_d > 0$, to model a detection error.

Proof. See Appendix B. □

Based on these results, Algorithm 3 is defined to generate realizations of the double Matérn cluster process. Note that in the double Matérn cluster process (Definition 4.6), the second generation points form metaclusters and the third generation points form clusters, while in the Neyman-Scott cluster process (Definition 4.5), the second generation points form clusters. The value of n_η in step 1 arises from a two-dimensional Taylor approximation carried out in order to calculate the expectation over the random variables S_c and S_{mc}, of which n_η cannot be derived as a linear function. For R_c, the distributional assumption $R_c \sim \text{Gamma}(r_c, 1)$ implies a higher variance for a higher r_c. For further theoretical details on the simulation algorithm, see Appendix B.

When point patterns are simulated according to Algorithm 3, the empirical clustering parameters in the point patterns will differ systematically from the simulated ones. The reason is that clusters of size zero, although they do count in terms of the proportion of clustered proteins and of the distribution of cluster sizes, are not manifested in the point pattern. Furthermore, clusters of

size one and singletons cannot be distinguished in a point pattern. For these reasons, one might argue that for the effects of the analyses in this thesis, the Poisson distribution has drawbacks in sampling the number of offspring points in a cluster. Three options exist: first, this could simply be ignored, since the estimation errors are calculated later based on the empirical clustering parameters of the point patterns, accepting that some of the values listed in Table 4.2 will not be met on average in the simulated point patterns. Second, one could try to calculate corrected input parameter values to cancel out the bias. Finally, a simple and small modification could be made to the simulation procedure in order to achieve that the simulated clustering parameter values are met on average by the resulting point patterns. Here, the last option is pursued, aiming to apply a simple modification to the Poisson distribution allowing to simulate only clusters of a certain size from the start. Clusters of size zero and one can be avoided in this way.

Therefore, instead of drawing the cluster sizes from a Poi(μ_c), i.e. a Poisson distribution with parameter μ_c, they are drawn from a Poi(μ_c-2) and subsequently 2 is added to the drawn values. While the shape of this distribution is different from a Poi(μ_c), the mean is again μ_c, which is important for the resulting point patterns to have the correct proportions of points in clusters and metaclusters (see Appendix B). However, the variance of this distribution is no longer μ_c, which means that the Taylor approximation carried out to calculate the number n_η of first generation points loses precision. Since this does not affect the fairness in the comparison of methods in a simulation study if the point density is not a quantity to be estimated (like in this thesis), the formula given in Section 4.2 is nevertheless used. Similarly, in order to avoid empty metaclusters as well as metaclusters that are not distinguishable from clusters, the metacluster sizes are drawn from a Poi(μ_c-2) and subsequently 2 is added to the drawn values. Potentially alternative modifications, such as omitting too small clusters during the simulation – discussed in van Lieshout and Baddeley (2002) in the context of avoiding empty clusters – appear unsatisfactory here

Algorithm 4 Simulating from the modified double Matérn cluster process

1. A number of n_η first generation points is sampled from a homogenous spatial Poisson process in Ψ,
2. A proportion $\tilde{p}_{mc}$ of first generation points is randomly replaced by metaclusters containing $S_{mc}+2$ second generation points distributed uniformly within a radius of $w_{mc} \cdot R_{mc}$, where $S_{mc} \sim \text{Poi}(\mu_{mc}-2)$, $\mu_{mc} > 1$, $w_{mc} \in \mathbb{R}$ and $R_{mc} \sim \text{Gamma}(r_{mc}, 1)$.
3. Of all points existent at this stage, a proportion of $\tilde{p}_c$ is replaced by clusters containing $S_c + 2$ third generation points distributed uniformly within a radius of $w_c \cdot R_c$, where $S_c \sim \text{Poi}(\mu_c - 2)$, $\mu_c > 1$, $w_c \in \mathbb{R}$ and $R_c \sim \text{Gamma}(r_c, 1)$.

To serve the needs of the application to Ras proteins, one may add:

4. Every remaining point $x \in \mathbb{R}^2$ is replaced by a point randomly drawn from $N(x, \sigma_d^2 \text{I})$, where $\sigma_d > 0$, to model a detection error.

since they alter the mean of the cluster size distribution.

In addition, note that the radius R_c^* of a cluster and the radius R_{mc}^* of a metacluster, when observed on simulated point patterns as one half of the maximal distance between any two points belonging to the same cluster or metacluster, will tend to be smaller than R_c and R_{mc} employed in steps 2 and 3 of Algorithm 3. Thus, to achieve point patterns in which $E(R_c^*) = r_c$ and $E(R_{mc}^*) = r_{mc}$, R_c and R_{mc} are multiplied with factors $w_c > 1$ and $w_{mc} > 1$, respectively, to remedy this drawback.

The algorithm taking into account these modifications is presented in Algorithm 4. To define the value of w_c and w_{mc} for each simulated pattern, a pre-simulation can be carried out employing $w_c = w_{mc} = 1$. For the final simulation, then $w_c = \frac{R_c}{R_c^*}$ and $w_{mc} = \frac{R_{mc}}{R_{mc}^*}$ are set, where R_c^* and R_{mc}^* are the values observed on the pre-simulation.

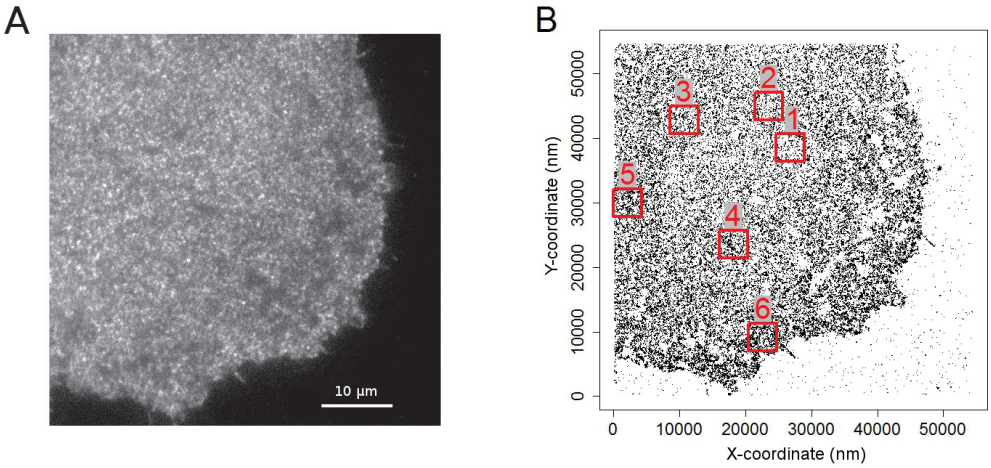

Figure 4.1: Images of a Ras-expressing cell. Shown are (**A**) total internal reflection fluorescence microscopy (TIRF) image of a cell expressing Ras tagged with mEos2 and (**B**) corresponding Ras localizations obtained by photo-activated localization microscopy (PALM) and preprocessing. Analyzed regions of interest 1-6 are marked.

4.3 Data

As a consequence of the experimental setup in PALM described in Chapter 2.2, the entire protein pattern of a cell is represented by a series of potentially noisy pixel images. Prior to a statistical analysis, these images have to be denoised, then protein coordinates have to be detected. Several methods have been proposed for each task, while a few methods offer a joint preprocessing pipeline for both. Many approaches use the information of several or all images jointly to track single points across the different time frames. A comparison of such methods is given in Cheezum *et al.* (2001). Here, the single-image approach rapidSTORM (Wolter *et al.*, 2010, 2012) is employed, consisting of three elements: all points in the image that do not represent local brightness maxima are disregarded, an average filter is applied to the remaining points, and Gaussian kernels with fixed covariance parameters are fitted to the these points. Due to measurement inaccuracies, drift, and light blob persistence, two protein locations detected within a range of five consecutive time frames and

parameter	symbol	values
overall proportion of points in clusters	p_c	0.2, 0.4, 0.6, 0.8
mean cluster size	μ_c	4, 8
mean cluster radius (nm)	r_c	15, 30
overall point density (points/μm^2)	Λ	50, 125, 200, 275
proportion of points in metaclusters	p_{mc}	0, 0.6, 0.8, 1
mean metacluster size	μ_{mc}	20, 30
mean metacluster radius (nm)	r_{mc}	100, 150
detection error (nm)	σ_d	0, 20

Table 4.2: Expected values of key parameters in the simulation study. Every possible combination of black values is considered concerning the respective other parameters. For gray values, only a subset of possible value combinations is considered concerning the other parameters: the cases $\Lambda = 50, 200, 275$ points/μm^2 are only simulated for $\mu_{mc} = 20$, $r_{mc} = 100$ nm and $\sigma_d = 20$ nm, and the cases $\mu_{mc} = 30$, $r_{mc} = 150$ nm and $\sigma_d = 0$ are only simulated for $\Lambda = 125$ points/μm^2. The combinations $\mu_{mc} = 20$, $r_{mc} = 150$ nm and $\mu_{mc} = 30$, $r_{mc} = 100$ nm are not considered.

within less than 25 nm physical distance are likely to correspond to a single protein. Thus, in such cases only one location is kept in the data set to avoid counting any protein multiple times.

Figure 4.1A shows a TIRF image of a cell membrane in which Ras proteins were tagged with the fluorescence protein mEos2, while Figure 4.1B shows the resulting protein coordinates detected by rapidSTORM as well as the location of several regions of interest (ROI) selected as a representative sample of typical protein densities. These ROIs are analyzed in Chapter 4.7, Figure 4.2A shows the coordinates of a ROI of this cell.

In experimental Ras data, the true clustering parameters of interest are unknown. An assessment of methods estimating them is thus difficult based on exclusively experimental data. For this reason, a comprehensive simulation study is conducted to compare the performance of such methods, complementing results on experimental data. The values that are considered for each parameter

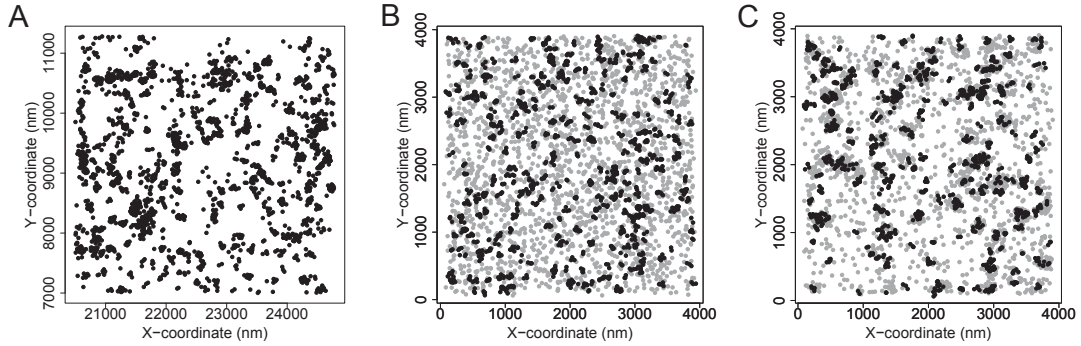

Figure 4.2: Ras localizations in region of interest 6 (as indicated in Figure 4.1) obtained by (**A**) PALM followed by preprocessing and localizations simulated with a point density of 200 points/μm^2, proportion of proteins in clusters $p_c = 0.4$, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 15$nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, with a proportion of proteins in metaclusters of either (**B**) $p_{mc} = 0$ or (**C**) $p_{mc} = 0.6$. Simulated proteins are shown either in gray (singletons) or black (clustered proteins).

in the simulation study are listed in Table 4.2. All simulated point patterns have an extension of 4 000 nm in square. The point densities of $\Lambda = 50, 200, 275$ points/μm^2 are only simulated in combination with the mean metacluster size $\mu_{mc} = 20$, the mean metacluster radius $r_{mc} = 100$ and a detection error of $\sigma_d = 20$ nm. For each simulation setting, five random replicates are generated and eventually the median results are calculated across replicates for comparison.

The modifications made in Algorithm 4 compared to Algorithm 3 result in mean point numbers per simulated point pattern of 832, 2 046, 3 295 and 4 535 for the expected point densities of 50, 125, 200 and 275 points/μm^2, respectively, i.e. the change leads to 2-4% more points per pattern (without the change, 800, 2 000, 3 200 and 4 400 points would be expected, respectively).

In Figure 4.2B and Figure 4.2C, point coordinates of simulated point patterns are shown in which a proportion of $\sim$ 40 % of proteins forms clusters located within circles with an average radius of 15 nm that contain an average of four proteins. In Fig. 4.2B no proteins form part of metaclusters, while in Fig. 4.2C $\sim$ 60 % of proteins do.

A visual comparison of the three point patterns shown in Figure 4.2 moti-

vates the simulation of the extra level of metaclustering as described in Section 4.2: the point pattern in Figure 4.2C involving metaclustering is structurally more similar to the experimental data (Figure 4.2A) than the point pattern without metaclustering shown in Figure 4.2B.

Point patterns are simulated using the `R` routine `rDoubleMatern` documented in the Appendix, Section C.1.

4.4 State-of-the-art Methods

In the following, three standard methods used to investigate clustering in the context of Ras proteins or in similar research are briefly described, including recent advances. They will be employed for performance comparisons later. For the remainder of Chapter 4, it is assumed that $\mathbf{X} = \{X_1, \ldots, X_n\}$ is a random point pattern in a compact region $\Psi \in \mathbb{R}^2$ corresponding to protein locations in a part of the cell membrane. The point coordinates are assumed to have undergone preprocessing such as noise filtering, as the discussed methods do not account for uncertainty in the point coordinates.

4.4.1 K-function Analysis

In Section 4.1.1, it is discussed how the presence of clustering in point patterns can be assessed employing the K-function $\mathcal{K}(r)$ or the related L-function $\mathcal{L}(r)$ and H-function $\mathcal{H}(r)$. A wide range of works about Ras protein clustering (e.g., Gurry *et al.*, 2009; Kiskowski *et al.*, 2009; Plowman *et al.*, 2005) employ the r value at which $\mathcal{H}(r)$ assumes its maximum as an estimate for the average cluster radius in a given point pattern. Alternatively, the minimum of the first derivative of $\mathcal{H}(r)$ has been considered as estimate (Kiskowski *et al.*, 2009).

As a drawback in the use of the K-, L- and H-functions when estimating the average radius of clusters, Definition (4.1) implies the intensity λ to be constant across the considered area Ψ. Precisely when clustering is assumed, this is not a reasonable assumption, and it is more sensible to generalize (4.1) in a way that

allows for λ to vary locally. This leads to the K-function for inhomogeneous point patterns (Baddeley *et al.*, 2000) that – additionally implementing an edge correction – is given by

$$\hat{\mathcal{K}}_{inh}(r) = \frac{1}{\sum_i \mathbf{I}(d_i^* > r)/\hat{\lambda}(x_i)} \sum_i \sum_{\mathrm{j} \neq i} \frac{1}{\hat{\lambda}(x_i) \cdot \hat{\lambda}(x_\mathrm{j})} \mathbf{I}(\|x_i - x_\mathrm{j}\| \leq r, d_i^* > r), \quad (4.7)$$

where $\hat{\lambda}(x_i)$ is the local intensity of the point process estimated at x_i, e.g., by a kernel smoother as described in Baddeley *et al.* (2000). To correct for edge effects, roughly, when estimating $\mathcal{K}_{inh}(r)$ only those points are considered that have a minimum distance of r to the border of the region of interest (ROI). Works in the biophysical community aiming to estimate parameters of Ras clusters mostly fail to employ (4.7) and inappropriately use (4.2).

The pair correlation function of the proteins has also been employed to achieve an estimate of the cluster radius. The PC-PALM algorithm (Sengupta *et al.*, 2011, 2013) fits an exponential function to the decay of the probability density of proteins from the center of a cluster and uses the estimated parameters of this function to derive a rough estimate for the cluster radius, but in addition also obtains estimates for the increased local density of proteins appearing in a cluster or domain and the average number of proteins in a cluster. By convolving with a point spread function for the uncertainty of the protein locations, the approach takes into account blinking.

Methods based on the K-function or derived functions mostly fail to estimate all three key clustering parameters (cluster radius, proportion of clustered proteins and cluster size). Some works that employ the H-function to estimate the mean cluster radius make assertions concerning cluster size and the proportion of clustered proteins as well, but for this end they rely on Monte Carlo tests simulating point patterns with many different possible parameters values on a fine grid, proceeding in a trial and error fashion in which lack of rejection of some values eventually replaces statistical estimation (see, e.g., Plowman *et al.*, 2005).

4.4.2 DBSCAN

DBSCAN (Ester *et al.*, 1996) is a well-established distance-based cluster algorithm. It implements the concept of density-based clustering, considering as cluster a set of points spread over a contiguous region of high point density. Density-based clusters are typically separated from each other by contiguous regions of low point density, and points located in such low-density regions are viewed as noise. Density-based clustering is related to single-linkage hierarchical clustering, which is often criticized for the 'chaining-effect', i.e. an undesirable connectivity of different clusters due to the existence of a 'chain' of single points between two clusters (Kriegel *et al.*, 2011). Density-based clustering was originally proposed by Wishart (1969) to remedy this drawback by removing low-density points before clustering the remaining points with single-linkage.

Technically, the definition of clusters is based on the concept of density-reachability w.r.t. two key parameters, ε and *MinPts*. A point ϱ_A is directly density-reachable from a point ϱ_B w.r.t. ε and *MinPts* if ϱ_B is within distance ε of ϱ_A, and if there are at least *MinPts* points lying in this ε-neighborhood of ϱ_A. Then, a point ϱ_A is density-reachable from a point ϱ_B w.r.t. ε and *MinPts* if a permutation $\pi(1), \ldots, \pi(n)$ of $1, \ldots, n$ exists for which in the chain $\varrho_A = x_{\pi(1)}, \ldots, x_{\pi(n)} = \varrho_B$, $x_{\pi(i+1)}$ is directly density-reachable from $x_{\pi(i)}$ w.r.t. ε and *MinPts*. Finally, two points ϱ_A and ϱ_B are density-connected w.r.t. ε and *MinPts* if there is a point ϱ_C such that both ϱ_A and ϱ_B are density-reachable from ϱ_C w.r.t. ε and *MinPts*. Density-connectivity, as opposed to density-reachability, is a symmetric condition. For given parameters ε and *MinPts*, DBSCAN defines a cluster as a set of density-connected points that is maximal in terms of density-reachability. Roughly, clusters are found by subsequently visiting all points and checking whether they are part of a set of density-connected points. All points of a point pattern not assigned to any cluster are defined as noise.

Data with a varying point density are problematic for DBSCAN, and depending on the choice of ε, the cluster detection may be biased in such cases.

The OPTICS algorithm (Ankerst *et al.*, 1999) seeks to improve the DBSCAN algorithm by reducing the dependence of its results on ε. The price for this is that no longer a cluster partition is obtained, but only a linear ordering of the points such that spatially closest points become neighbors in the ordering. The output can therefore be visualized in form of a dendrogram, like in hierarchical clustering.

Nan *et al.* (2013) propose a variant named simulation-aided DBSCAN that simulates data sets containing the same amount of noise inherent to protein locations obtained from a PALM image. It analyzes both the simulated and the experimental data set with DBSCAN and compares the results. Simulation parameters are iteratively adjusted until a simulated data set is produced on which DBSCAN results match for both the simulated and the PALM data.

In each of these algorithms, the proportion of clustered proteins as well as the mean cluster size and radius can be derived from the obtained cluster partition. In this thesis, the original DBSCAN algorithm is employed.

Since the performance of DBSCAN was heavily dependent on the choice of the parameter ε in preliminary analyses on simulated data, two different values are considered. One value, $\varepsilon = 20$ nm, is close to the true cluster radii in the simulation study and, presumably, in experimental Ras data, while the other value ($\varepsilon = 100$ nm) is much larger than the true cluster radii. Following the general explicit discrimination between singletons and clusters in this thesis, *MinPts* is set to two.

4.4.3 Model-Based Clustering

In model-based clustering as described in Section 3.3, observations that supposedly manifest grouping or clustering are modeled by a mixture of distributions, most often Gaussian ones. In spatial cluster analysis, models with a (potentially) infinite number of components or approximations to such models are common for a variety of applications. Dasgupta and Raftery (1998), e.g., design

a mixture model for the detection of spatial clusters with a long and narrow shape in presence of noise and use the EM algorithm to fit it to data about minefields and seismic faults. Kottas and Sansó (2007) apply a DPM prior to cluster tree seedlings, and Ji *et al.* (2009) employ a truncated Dirichlet process to cluster cells in a tissue. Ren *et al.* (2011) introduce the class of logistic stick-breaking processes and demonstrate their utility for the segmentation of images into homogenous parts. Gelfand *et al.* (2007) give an overview of nonparametric mixture models using Dirichlet processes applied to spatially referenced data. The very popular DPM model has been widely used to analyze clustering. It is formulated as follows:

$$X_i|\tilde{\mu}_i, \Sigma_i \sim N(\tilde{\mu}_i, \Sigma_i), i = 1, \ldots, n,$$

$$(\tilde{\mu}_i, \Sigma_i)|\Psi \sim G \quad \text{with} \quad G|\alpha_0, G_0 \sim DP(\alpha_0, G_0),$$

assuming a conjugate normal-inverted Wishart as baseline distribution,

$$G_0 \sim N(\tilde{\mu}|m_1, (1/k_0)\Sigma)IW(\Sigma|\nu_1, \psi_1),$$

and with hyperpriors

$$\begin{aligned} \alpha_0|a_0, b_0 &\sim \text{Gamma}(a_0, b_0), \\ m_1|m_2, s_2 &\sim N(m_2, s_2), \\ k_0|\tau_1, \tau_2 &\sim \text{Gamma}(\tau_1/2, \tau_2/2), \\ \psi_1|\nu_2, \psi_2 &\sim IW(\nu_2, \psi_2). \end{aligned}$$

When fitting this model employing Markov Chain Monte Carlo (MCMC) algorithms, a cluster partition is obtained in each iteration. However, sensible posterior averaging is not straightforward across entire partitions. The most intuitive way to obtain a single posterior cluster partition would therefore be to average across the posterior distributions of the cluster allocation variables for the $x_i, i = 1, \ldots, n$. This is however complicated by unstable estimates due to

label switching, as discussed in Section 3.3. One solution is to obtain a posterior similarity matrix instead, which exhibits a low susceptibility to label switching. A cluster partition may then be found by optimizing a criterion based on this matrix, such as Binder's loss (Binder, 1978).

Since in this thesis the interest lies in estimating clustering parameters rather than finding a single partition, one could calculate the clustering parameters (proportion of clustered points, cluster radius, cluster size) in each MCMC iteration and afterwards aggregate across them. However, while this would be ideal in the Bayesian spirit, in this thesis better results are achieved by first obtaining a single partition based on the posterior similarity matrix and estimating the parameters of interest from it.

Argiento *et al.* (2013) aim to reduce the instability of cluster estimates in the DPM model. For this purpose, they relax the natural clustering rule which groups points assigned to the same kernel densities. Specifically, they group points assigned to densities which have a mutual distance of ε or less in an appropriate metric for densities. To accomplish this, they run the DBSCAN algorithm on the densities sampled in each of the DPM model's MCMC iterations. Eventually, again one cluster partition is estimated by optimizing a criterion such as Binder's loss. The authors suggest to do this for a grid of several ε values and to choose the best one in terms of the criterion. To measure the distances between Gaussian kernels, the Kullback-Leibler, Hellinger and L^2 distances are considered. The method proposed by Argiento *et al.* (2013), henceforth referred to as Bayesian DBSCAN (bDBSCAN), is similar to GAMMICS regarding its motivation to embed DBSCAN in a Bayesian framework. It will be employed for some exemplary comparisons.

As the performance of the DPM model on simulated data in preliminary analyses was heavily dependent on whether informative or non-informative priors are chosen, two prior configurations are chosen for it, one with informative priors and one with non-informative priors. As informative priors, $m_{2,0} = 0$, $s_{2,0} = \psi_{2,0} = I$, $a_{0,0} = 0.1$, $b_{0,0} = 0.1$, $\nu_{1,0} = 50$, $\nu_{2,0} = 4$, $\tau_{1,0} = 0.1$ and $\tau_{2,0} = 0.1$

are chosen, while as non-informative priors, $m_{2,0} = 0$, $s_{2,0} = \psi_{2,0} = I$, $a_{0,0} = 1$, $b_{0,0} = 1$, $\nu_{1,0} = 4$, $\nu_{2,0} = 4$, $\tau_{1,0} = 1$ and $\tau_{2,0} = 1$ are chosen.

By contrast to the DPM model, the implementation of the Bayesian DBSCAN method does not enable the specification of additional hyperpriors. As informative priors, thus, $a_0 = 30.25$, $b_0 = 2.75$, $\nu_{1,0} = 1\,000$, $k_{0,0} = 0.001$ and $\psi_{1,0} = 0.1I$ are chosen directly for Bayesian DBSCAN. In preliminary analyses, it was shown that non-informative priors similar to those considered for the DPM model, e.g., $a_0 = 1$, $b_0 = 1$, $\nu_{1,0} = 4$, $k_{0,0} = 1$ and $\psi_{1,0} = I$, generally led to insensible results and frequent early breakdowns of the program due to all points being allocated to one cluster during sampling. Thus, they are not further considered for Bayesian DBSCAN. The value of ε, of little impact on the results for the parameters of interest in preliminary analyses, is fixed to $\varepsilon = 0.005$ nm based on the scaling of the data region (see also Section 4.7).

4.5 The GAMMICS Method

4.5.1 Model

The main goal of the statistical analysis of Ras protein patterns in this thesis is to estimate the proportion of clustered proteins, the mean cluster size and the mean cluster radius as key parameters of Ras nanoclustering. Building a model for this purpose is challenging. In particular, jointly modeling the cluster size and the cluster radius turns out to be difficult because for any given cluster in the point pattern, the two measures are not independent. Rather, given the center and either the size or the radius of the cluster, the value of the remaining measure can be derived from the point pattern, respectively. Thus, a strictly probabilistic model for the point pattern would encounter problems in identifying all three parameters of interest simultaneously. The cluster size and the cluster radius, or at least one of them, would have to be fixed.

The solution pursued instead in Schäfer *et al.* (2015) and described here

in more detail is to not fit a model to the point pattern itself, but rather to distinguish clustered points and singletons by using the points' distances to their κth nearest neighbors. The reasoning is that these distances may differ structurally between clustered and non-clustered points: for any point that lies in a cluster, the distance to its κth nearest neighbor can be expected to be smaller than for any singleton. Diggle (2003) and Wiegand *et al.* (2009) point out that nearest neighbor characteristics are an appropriate means to distinguish a point pattern arising from a Poisson cluster process from a point pattern arising from the superposition of a (homogenous) Poisson process and a Poisson cluster process. The idea of employing distances to κth nearest neighbors has been described by Byers and Raftery (1998) in the context of distinguishing large objects from clutter. For an optimal separation of objects and clutter, they recommend to choose κ as about the size of the smallest feature one wishes to detect. Following this argument, in the context of small Ras protein clusters that imply a minimal object (cluster) size of two, it appears reasonable to choose $\kappa = 2$.

Let the random variables $D_i(X_i; \mathbf{X} \backslash X_i)$, $i = 1, \ldots, n$, represent the Euclidean distances between points X_i and their respective second nearest neighbor. The notation $D_i(X_i; \mathbf{X} \backslash X_i)$, hereafter written D_i, intends to reflect the fact that each distance depends not only on the location of the corresponding X_i, but also on all other points, i.e. on the whole $\mathbf{X}$. Byers and Raftery (1998) propose and theoretically justify to fit gamma distributions to the square of such distances D_i when aiming to distinguish objects from clutter in the context of superimposed Poisson processes. This argument is picked up in the following, fitting a mixture of two gamma distributions to realizations of the squared distances $\boldsymbol{D^2} = (D_1^2, \ldots, D_n^2)$ between points and their respective second nearest neighbors. One of the two gamma distributions is intended to represent the D_i^2 corresponding to clustered proteins, the other one is intended to represent the D_i^2 corresponding to singletons.

While the proportion of clustered proteins can be estimated as a model

parameter in such a framework, the cluster size and the cluster radius are estimated via a partly algorithmic approach based on information originating from the model. This approach allows to obtain estimates of all three parameters of interest.

Binary variables allocating points to the two mixture components are defined as

$$T_i(D_i^2) = \begin{cases} 1 & \text{if } D_i^2 \text{ corresponds to a clustered } X_i, \\ 0 & \text{if } D_i^2 \text{ corresponds to a non-clustered } X_i, \end{cases} \tag{4.8}$$

for $i = 1, \ldots, n$. The shape and rate parameters of the two gamma distributions are in turn assigned gamma hyperpriors, respectively,

$$D_i^2 | T_i = k \sim \text{Gamma}(\tilde{\alpha}_k, \tilde{\beta}_k) \quad \text{for} \quad k = 0, 1, \tag{4.9}$$

$$\tilde{\alpha}_k | T_i = k \sim \text{Gamma}(a_{\tilde{\alpha}_k}, b_{\tilde{\alpha}_k}) \text{ for} \quad k = 0, 1, \tag{4.10}$$

$$\frac{1}{\tilde{\beta}_k} \Big| T_i = k \sim \text{Gamma}(c_{\tilde{\beta}_k}, d_{\tilde{\beta}_k}) \text{ for} \quad k = 0, 1, \tag{4.11}$$

for $i = 1, \ldots, n$, where $p(x) = \frac{1}{\Gamma(\alpha)\beta^\alpha} x^{\alpha-1} e^{-\frac{x}{\beta}}$ for $x, \alpha, \beta > 0$ is the density of the Gamma(α, β) distribution, parameterized in terms of the shape parameter α and the scale (inverse rate) parameter β.

When fitting the model, the points $X_i, i = 1, \ldots, n$, are assigned to one of the gamma distributions by means of the allocation vector $\boldsymbol{T} = (T_1, \ldots, T_n)$, where each allocation variable T_i is following a binomial distribution. The mixture weight in turn is following a beta distribution,

$$T_i \sim \text{Bernoulli}(p_c), \tag{4.12}$$

$$p_c \sim \text{Beta}(a_{p_c}, b_{p_c}), \tag{4.13}$$

for $i = 1, \ldots, n$. Jointly, equations (4.8)-(4.13) constitute the mixture model underlying the GAMMICS method.

The a priori hyperparameters for GAMMICS are chosen as $a_{\tilde{\alpha}_0,0} = 2$, $b_{\tilde{\alpha}_0,0} = 1$, $c_{\tilde{\beta}_0,0} = 10$, $d_{\tilde{\beta}_0,0} = 1$ (distribution covering the singletons), $a_{\tilde{\alpha}_1,0} = 3$, $b_{\tilde{\alpha}_1,0} = 1$,

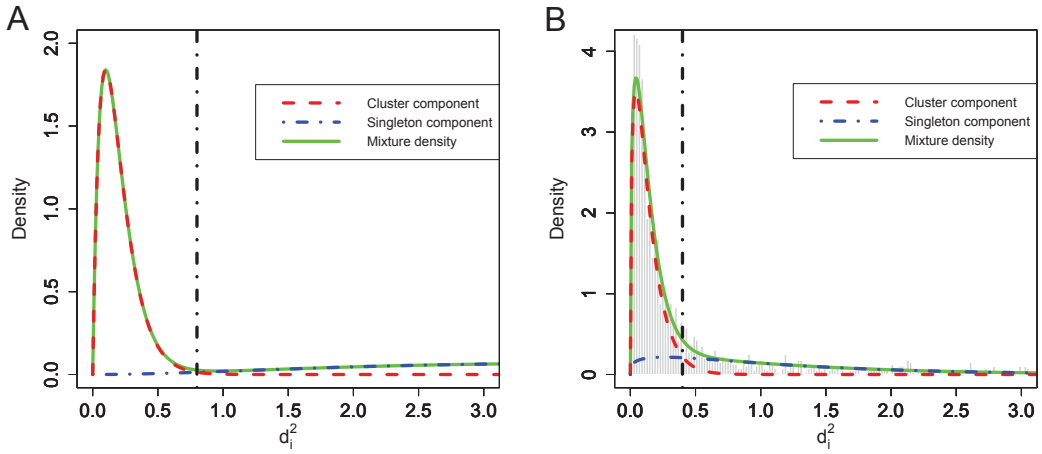

Figure 4.3: Fitting gamma distributions to (normalized) squared distances between points and their second nearest neighbors. Density functions of the two gamma distributions as well as the overall mixture density are plotted according to (**A**) the prior parameter values and (**B**) the posterior parameter values obtained for a point pattern simulated with a proportion $p_c = 0.4$ of proteins in clusters, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 30$ nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, a proportion of proteins in metaclusters of $p_{mc} = 0.6$ and a detection error of $\sigma_{DE} = 20$ nm. The intersection of the two gamma densities is marked with a dot-dashed vertical line.

$c_{\tilde{\beta}_1,0} = 1$, $d_{\tilde{\beta}_1,0} = 0.5$ (distribution covering the clustered points) and $a_{p_c,0} = b_{p_c,0} = 1$ (uniform prior distribution for p_c). In preliminary analyses, these values have led to a reasonably good separation of clustered and non-clustered points. The prior probability density functions for the two gamma distributions corresponding to these choices are shown in Figure 4.3A. In Figure 4.3B, two gamma distributions fitted to a histogram of (normalized) squared distances between points of a simulated point pattern and their second nearest neighbors are shown.

4.5.2 Algorithmic Step to Estimate Cluster Sizes and Cluster Radii

To be able to infer the mean cluster size and the mean cluster radius, the classification to one of the two groups (clustered vs. non-clustered) obtained

Algorithm 5 Estimation of cluster partition in GAMMICS iteration

1. Select one d_c^2 among the observed $d_i^2, i = 1, \ldots, n$, as cutoff between the d_i^2 belonging to clustered and the d_i^2 belonging to non-clustered points,

$$d_c^2 = \begin{cases} \min\{d_i^2 | d_i^2 \geq d_{m,0}^2 \wedge \delta_0(d_i^2) < 0\} & \text{if } d^+ = \varnothing \vee d^- = \varnothing, \\ \min\{d_i^2 | d_i^2 \geq d_m^2 \wedge \delta(d_i^2) < 0\} & \text{otherwise,} \end{cases} \tag{4.14}$$

   where d_m^2 ($d_{m,0}^2$) is the d_i^2 with highest probability according to the gamma density with more probability mass for small distances in the current MCMC iteration (in the prior configuration),

$$d_{m0}^2 = \arg\max_{d_i^2} p(d_i^2 | \tilde{\alpha}_{1,0}, \tilde{\beta}_{1,0}), \tag{4.15}$$

$$d_m^2 = \arg\max_{d_i^2} p(d_i^2 | \hat{\tilde{\alpha}}_{\Delta}, \hat{\tilde{\beta}}_{\Delta}), \tag{4.16}$$

$$\Delta = \mathbf{I}\left(median\{d^+\} < median\{d^-\}\right), \tag{4.17}$$

   where $\mathbf{I}()$ is the indicator function and 'small distances' are defined via the medians of two sets of d_i^2: each set contains the d_i^2 for which one of the two weighted gamma densities gives higher probability,

$$d^+ = \{d_i^2 | \delta(d_i^2) > 0\}, \quad d^- = \{d_i^2 | \delta(d_i^2) < 0\} \tag{4.18}$$

$$\delta(x) = \hat{p}_c \cdot p(x | \hat{\tilde{\alpha}}_1, \hat{\tilde{\beta}}_1) - (1 - \hat{p}_c) \cdot p(x | \hat{\tilde{\alpha}}_0, \hat{\tilde{\beta}}_0), \tag{4.19}$$

$$\delta_0(x) = p(x | \tilde{\alpha}_{1,0}, \tilde{\beta}_{1,0}) - p(x | \tilde{\alpha}_{0,0}, \tilde{\beta}_{0,0}), \tag{4.20}$$

   for $i = 1, \ldots, n$, and $x \in \mathbb{R}^+$. $\hat{\tilde{\alpha}}_0$, $\hat{\tilde{\beta}}_0$, $\hat{\tilde{\alpha}}_1$, $\hat{\tilde{\beta}}_1$ and $\hat{p}_c$ are the estimates for $\tilde{\alpha}_0$, $\tilde{\beta}_0$, $\tilde{\alpha}_1$, $\tilde{\beta}_1$ and p_c in the current MCMC iteration, while $\tilde{\alpha}_{0,0}$, $\tilde{\beta}_{0,0}$, $\tilde{\alpha}_{1,0}$ and $\tilde{\beta}_{1,0}$ are corresponding a priori expected values. Typically, d_c^2 will approximate the intersection between the density functions of the two fitted gamma distributions.

2. Calculate the distance $\tilde{d}_c$ of point X_c (corresponding to d_c^2) to its nearest neighbor.

3. Apply single-linkage hierarchical clustering to all points classified as clustered until all distances between groups are greater than $\tilde{d}_c$.

from the mixture model is not sufficient. In addition, the allocations of points to specific clusters are needed. For this reason, the cluster partition is deduced in a post-processing step employing Algorithm 5.

In step 1, a cutoff between the population of clustered points and singletons

is calculated based on the two gamma densities fitted to the squared distances of points to their second nearest neighbors. In step 2 and step 3, on the other hand, the cluster partition is estimated based on a cutoff for the ordinary distances between points. A central idea of Algorithm 5 is thus to identify an observation representing the cutoff, since both types of distance can then be easily calculated for this observation.

The algorithm assigns any two points to the same cluster if they have a mutual distance of $\tilde{d}_c$ or less. Suppose that, e.g., in Fig. 4.4, $\mathfrak{O}$, $\mathfrak{E}$ and $\mathfrak{C}$ are points and $\mathfrak{o} \leq \tilde{d}_c$, $\mathfrak{e} \leq \tilde{d}_c$, but $\mathfrak{c} > \tilde{d}_c$. Then, although $\mathfrak{O}$ and $\mathfrak{E}$ lie further apart than $\tilde{d}_c$, all three points will be defined to be part of one cluster, since $\mathfrak{O}$ and $\mathfrak{C}$ as well as $\mathfrak{E}$ and $\mathfrak{C}$ are assumed to be in the same cluster, respectively. This is typical for single-linkage clustering. By consequence, points in a detected cluster may have a distance larger than $\tilde{d}_c$, as long as there are enough other points lying between them so that no distance between nearest neighbors exceeds $\tilde{d}_c$. This algorithmic design helps to avoid limiting prior assumptions w.r.t. the shapes of the clusters. Requiring all proteins in one cluster to be no further than $\tilde{d}_c$ apart would tend to limit the sensitivity of the algorithm to circularly shaped clusters. Clusters with a prolate or irregular shape, displaying a greater variance of nearest neighbor distances within the cluster, would not be detected with an equally high probability. Step 3 of Algorithm 5 is equivalent to applying DBSCAN (Ester *et al.*, 1996) with parameters $\varepsilon = \tilde{d}_c$ and $MinPts = 2$ to all points classified as clustered by the mixture model. By definition, no clusters of size one are produced. Altogether, GAMMICS may be viewed as a Bayesian version of established clustering algorithms, where the crucial parameter (the dendrogram cutoff in hierarchical clustering, or ε in DBSCAN) is estimated by a Bayesian mixture model.

The definition of d_c^2 in formulas (4.14)-(4.20) has a crucial influence on the estimates for cluster size and radius. In (4.19), the alternative definition $\delta(x) = p(x|\hat{\tilde{\alpha}}_1, \hat{\tilde{\beta}}_1) - p(x|\hat{\tilde{\alpha}}_0, \hat{\tilde{\beta}}_0)$ might also be used, omitting the weights p_c and $1 - p_c$. This is equivalent to setting $\hat{p}_c = 0.5$ when calculating $\delta(x)$ via (4.19). While

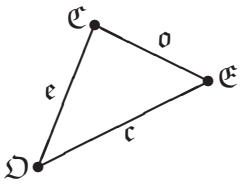

Figure 4.4: Artificial example of three point localizations $\mathfrak{O}$, $\mathfrak{E}$ and $\mathfrak{C}$.

introducing a bias in cases where $|\hat{p}_c - 0.5|$ is large, this definition may help to robustify the cutoff against artificially small or large estimates of p_c. It will also be considered later for this reason. To define d_c^2, one could further consider quantiles of those d_i^2 which are assigned to one of the two gamma components in the model.

4.5.3 Sampling and Robustness Measures

Whenever possible, the priors in the GAMMICS model, i.e. in equations (4.8)-(4.13), have been assigned so that the full conditionals are conjugate. Using MCMC methods, the model can therefore be easily fitted with a Gibbs sampling scheme. For the shape parameter of the gamma distributions in (4.10), a Metropolis-Hastings step is included since no conjugate prior exists. A normal proposal distribution centered at the actual value and mirrored at 0, with a reasonably tuned standard deviation σ_{MH}, is found to achieve an appropriate acceptance rate. For the sake of mathematical simplicity, it is assumed that the $D_i^2, i = 1, \ldots, n$, and thus the $T_i, i = 1, \ldots, n$, are i.i.d. (this assumption is not without problems, see also Discussion). The joint likelihood of all parameters and corresponding full conditionals as well as the full sampling scheme for the Gibbs Sampler can be found in Section A.2 in the Appendix.

To achieve a more robust estimation against outliers, in each MCMC iteration the radius of each cluster is calculated as the mean pairwise distance between points within the cluster, multiplied with the factor $2 \cdot \frac{45\pi}{128}$. This fac-

tor corresponds to the relation between the mean and the maximum in the theoretical distribution of the distance between two random points distributed uniformly inside the unit circle (see the Appendix, Remark B.1 for more details). The median estimation error is found to be much smaller in this way than by using, e.g., one half of the maximal distance between any two points belonging to the same cluster as estimator (the latter measure was employed in the simulation study, see Section 4.3).

In each MCMC iteration, the application of Algorithm 5 leads to a distribution of cluster radii and cluster sizes across all clusters in the analyzed point pattern. Thus, in addition to the respective mean values across the clusters, it is possible to record also of a number of quantiles for each iteration. Eventually, a posterior distribution across the clusters can be represented using the resulting mean posterior quantiles.

To avoid numerical problems when applying GAMMICS to data sets that present distances of different average magnitude, the observed d_i^2 are normalized for the model fitting. Specifically, they are divided by twice their median absolute deviation from the median.

Further potential sources of bias are outliers in the empirical distribution of the $d_i^2, i = 1, \ldots, n$, which may arise, e.g., due to edge effects. When clusters are spatially divided by the limits of the analyzed region Ψ, it may happen that some clustered points falsely appear as singletons within Ψ, potentially affecting the fit of the model and the resulting estimates. To avoid this, a larger region Ψ^* encompassing Ψ is defined by adding a border strip $\mathcal{B}$ with a predefined width w at the limits of Ψ, so that $\Psi^* \backslash \mathcal{B} = \Psi$. The points lying in $\mathcal{B}$ are only used to calculate the d_i^2 for the points in the inner region Ψ, but are not considered within the model itself. Points within Ψ classified as clustered only due to points lying in $\mathcal{B}$ are removed from the list of clustered points. As a further measure to prevent harmful outlier influence, the observed sample of $d_i^2, i = 1, \ldots, n$, is trimmed prior to the analysis. Specifically, all d_i^2 exceeding the difference between $q_{0.975}$ and $q_{0.025}$, the 0.975 and 0.025 quantiles of the

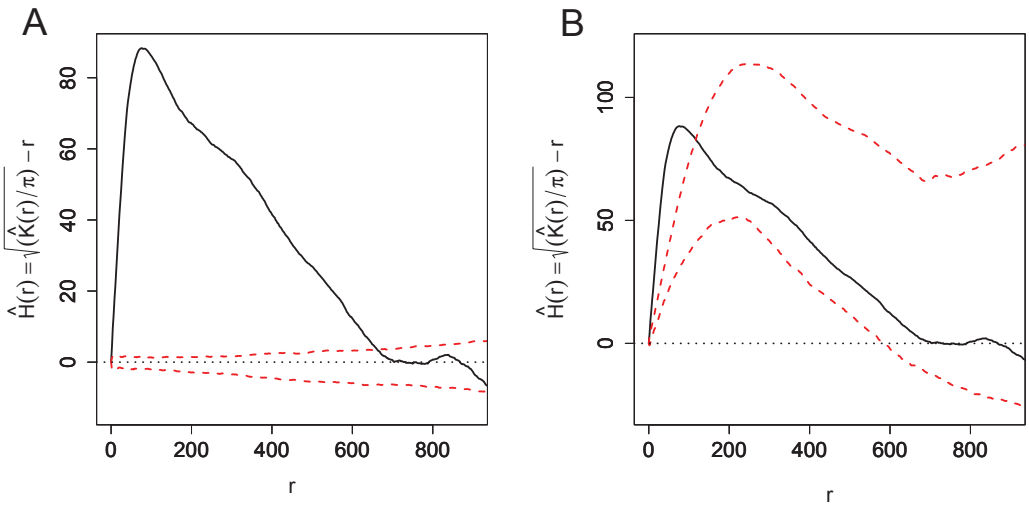

Figure 4.5: Visualization of Monte Carlo tests with 500 iterations for cluster proportions of (**A**) $p_c = 0$ and (**B**) $p_c = 1$ with $p_{mc} = 0.5$, $\mu_{mc} = 20$, $r_{mc} = 100$ nm, for ROI 6 of the example data as shown in Figure 4.2A. The solid line corresponds to the H-function calculated for the data, the 99% pointwise confidence bands are marked with dotted lines.

empirical distribution of the d_i^2, are not taken into account in the model.

4.6 Monte-Carlo Tests for GAMMICS

The GAMMICS method relies mainly on fitting a mixture of two distributions to an empirical distribution of distances, and dividing the distances into two sets corresponding to clustered and non-clustered points is essential. For this reason, GAMMICS has weaknesses in detecting the extreme cases of $p_c = 0$ and $p_c = 1$, i.e. the situations of no clustering or all points being clustered, respectively. Although in these cases there is no sensible cutoff to distinguish two groups of proteins, the method will nevertheless tend to estimate one.

A reasonable strategy to rule out the cases $p_c = 0$ and $p_c = 1$ can be based on Monte Carlo tests. Although the H-function $\mathcal{H}(r)$ as described in Section 4.4.1 may have limitations in estimating clustering parameters, it is assumed to be sufficiently sensitive for these extreme cases to be used as measure in

Algorithm 6 GAMMICS analysis including Monte-Carlo tests for p_c

1. Conduct a Monte Carlo test for the case $p_c = 0$.
2. Obtain estimates for cluster size and radius for the case $p_c = 1$, e.g., by running DBSCAN with $MinPts = 2$ and ε as the maximum nearest neighbor distance present in the data set.
3. Using the extra estimates for the case $p_c = 1$, conduct a Monte Carlo test for this case. As the model does not estimate the parameters of metaclustering (p_{mc}, μ_{mc}, r_{mc}), the test has to be carried out repeatedly for a reasonable range of those parameters.
4. If the Monte Carlo test rejects both the hypotheses of no clustering and of 100% clustering, run the GAMMICS method.

corresponding Monte Carlo tests. Monte Carlo tests for the K-function and derived functions have been described as a means to test for complete spatial randomness (Besag and Diggle, 1977), but also for more complex point processes (Dixon, 2002). Stoyan (1992), e.g., discusses results that allow to perform Monte Carlo tests for the Matérn cluster process. As an alternative to the H-function, the pair correlation function could also be used. Mahling *et al.* (2013), e.g., employ both the K-function and the pair correlation function for Monte Carlo tests on point patterns with potential clustering. To carry out a Monte Carlo test in the context of Ras proteins, point patterns are simulated by employing Algorithm 4 in Section 4.2. Note that to test for $p_c = 1$, the simulated cluster sizes and radii should not differ considerably from the true ones of the tested point pattern, otherwise the test result may be erroneous. Thus, it is essential to obtain estimates for cluster size and radius conditional on $p_c = 1$ before carrying out the test. For this purpose, e.g., DBSCAN can be run using the `fpc` package, specifying $MinPts$=2 and setting ε to the maximum nearest neighbor distance present in the data set. Since no estimates are obtained for the parameters of metaclustering (p_{mc}, μ_{mc}, r_{mc}), the test has to be carried out repeatedly for a reasonable range of those parameters.

The Monte Carlo test allows to calculate confidence bands for the null dis-

tributions across a relevant range of r values. This is visualized in Figure 4.5 for an example. If the H-function calculated from the data at hand lies inside the confidence band plotted for the case $p_c = 0$ (Figure 4.5A), it can be concluded that no clustering is present. If, on the other hand, this H-function lies inside the confidence band corresponding to the case $p_c = 1$ (4.5B), it can be concluded that all proteins form part of clusters. In both cases, fitting the mixture model underlying GAMMICS would not lead to sensible results. In Figure 4.5, however, it can clearly be seen that in this case the tests reject both the cases $p_c = 0$ and $p_c = 1$.

A full algorithmic scheme for the entire analysis including the Monte Carlo tests is given in Algorithm 6.

4.7 Results

In this section, the performance of the GAMMICS method is assessed regarding its performance in estimating key parameters of Ras clustering. For this purpose, GAMMICS is compared with the competing approaches described in Section 4.4: the H-function $\mathcal{H}(r)$, DBSCAN, a DPM model, and the Bayesian DBSCAN method developed by Argiento *et al.* (2013), which is a mixture of the latter two. The H-function is employed only to estimate the mean cluster radius.

Markov chains for GAMMICS are run with 23 000 iterations, where the first 8 000 iterations are discarded as burn-in and every 30th of the remaining values is used for the analysis in order to reduce autocorrelation. Based on trace plot examination, this ensures stability and convergence of Markov chains (exemplary trace plots are shown in the Appendix, Section A.4). For the standard deviation of the proposal distribution in the Metropolis-Hastings steps for sampling the gamma shape parameters $\tilde{\alpha}_0$ and $\tilde{\alpha}_1$, $\sigma_{MH} = 0.1$ is chosen, leading to median acceptance rates of 0.374 for $\tilde{\alpha}_0$ and 0.262 for $\tilde{\alpha}_1$. The method is run on MATLAB, version 7.10.0 (MATLAB, 2010), while additional comput-

ing is performed in `R`, version 3.1.0 (R Core Team, 2014). The MATLAB code implementing GAMMICS is documented in Appendix C.3.

DBSCAN is run employing the `R` package `fpc` (Hennig, 2014), using two different parameter configurations to demonstrate how the method performs with a favorable and with an unfavorable choice of the key parameter ε.

The DPM model is applied by means of the `R` package `DPpackage` (Jara *et al.*, 2011). The model is fitted twice to each point pattern, once with an informative prior configuration and once with a non-informative prior configuration. Each run consists of 15 000 iterations, where every 10th value is used for analysis after a previous burn-in of 1 000 iterations. Subsequently, using the `R` package `mcclust` (Fritsch and Ickstadt, 2009), the posterior similarity matrix is calculated. A cluster partition is chosen using Binder's loss, since this criterion results in smaller estimation errors than the maximum expected adjusted Rand index on simulated data. To find the partition minimizing Binder's loss, different methods are considered: first, the posterior similarity matrix is subtracted from 1 for conversion to a distance matrix, and hierarchical clustering is carried out with average or complete linkage, calculating the posterior expected loss clusterings for different cutoffs. Additionally, the clusterings obtained in the different MCMC iterations are compared regarding their posterior expected loss. Finally, the partition with the smallest loss among all considered partitions is selected.

The Bayesian DBSCAN method by Argiento *et al.* (2013) is applied using a series of C and `R` files published by the authors. Each run consists of 75 000 iterations, where every 15th value is used for analysis after a previous burn-in of 5 000 iterations. When fitting Gaussian kernels to the many small clusters in the data considered for this thesis, differences in mean matter presumably more than differences in covariance matrices. As Argiento *et al.* (2013) affirm, in such a setting the L^2 distance is better suited than the Kullback-Leibler or Hellinger distances. It is therefore employed here. For both the DPM model and Bayesian DBSCAN, the data are rescaled to a unit square centered at 0

before the model is fit, and the clustering parameters are calculated based on one final partition estimated by the model (clusters of size one are not counted as clusters).

In GAMMICS, points are classified as clustered or as singleton by voting across MCMC iterations. To measure the misclassification rate, in all methods a border strip $\mathcal{B}$ is excluded at the limits of the data region P^* for comparability, since GAMMICS does not consider it. The estimates for μ_c and r_c are obtained from cluster partitions in different ways. For DBSCAN, they are calculated in the same way as for GAMMICS (see Section 4.5.3). For the DPM model and Bayesian DBSCAN, the median across clusters is calculated and used as estimate, the radius of one cluster being calculated as one half of the maximum distance between any two points belonging to the cluster. These choices lead to the best results for each method. Aggregated results when employing an alternative way of estimating the cluster radius or when aggregating with the mean instead of the median (or vice versa) are documented in Tables A.6, A.7, A.8 and A.9 in Appendix A.4. When two variants are considered for a method (e.g., $\varepsilon = 20$ nm or $\varepsilon = 100$ nm for DBSCAN), the better performing one decides the choice of the estimator. For a given method, cluster sizes and cluster radii are either both aggregated via the mean or both aggregated via the median.

For the H-function estimations, the `R` package `spatstat` (Baddeley and Turner, 2005) is used. In case of multiple local maxima of $\mathcal{H}(r)$, the one assumed at a smaller r is selected.

4.7.1 Simulated Data

Aggregated misclassification rates and absolute estimation errors achieved by the considered methods in the simulation study described in Section 4.3 are reported in Fig. 4.6. Results are presented in dependence of p_c, μ_c, r_c and p_{mc}, calculating the median across simulation scenarios and MCMC iterations. Further aggregated results in dependence of Λ, μ_{mc}, r_{mc} and σ_d are reported for

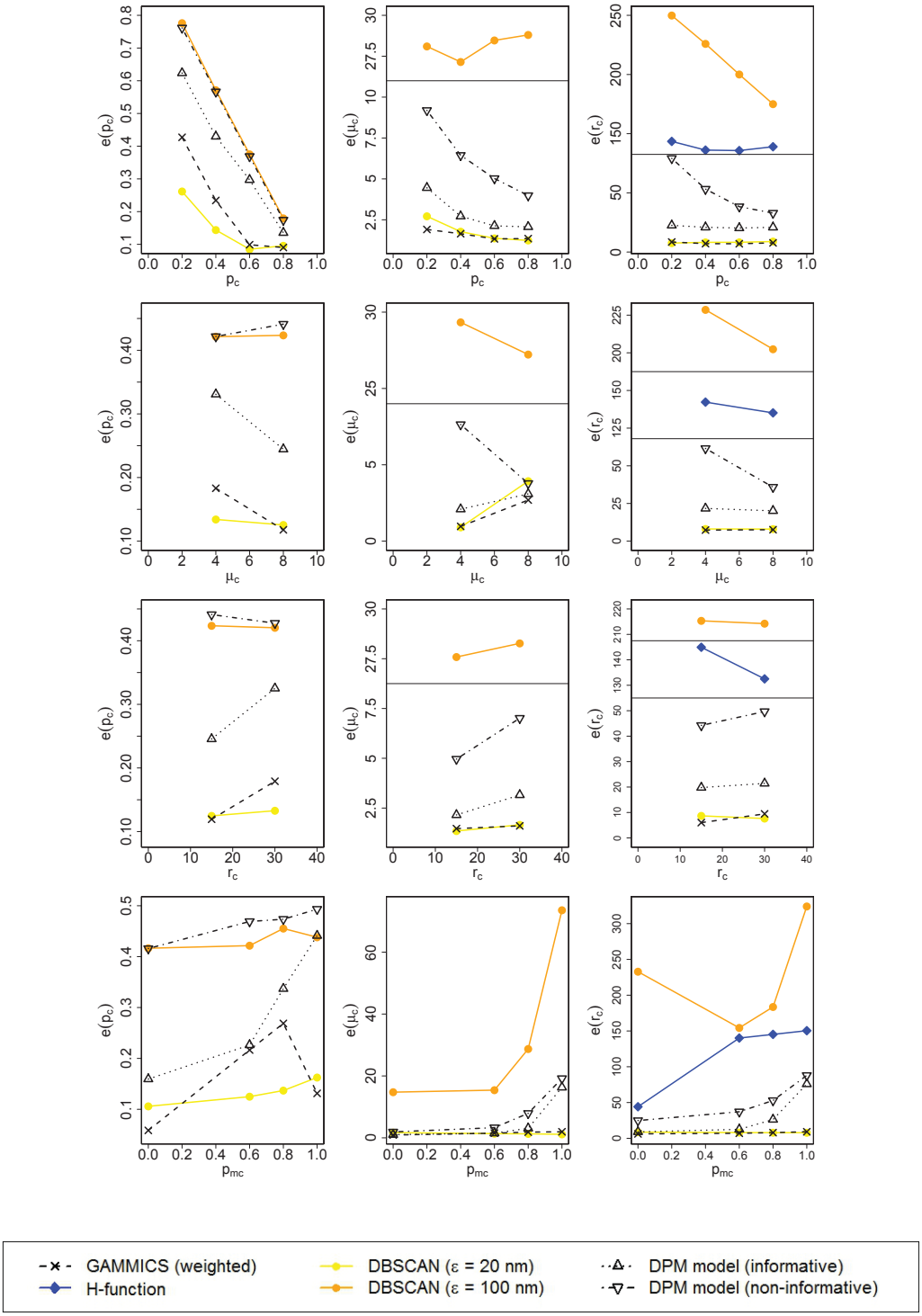

Figure 4.6: Simulation results. Median absolute estimation errors $e(\cdot)$ for the expected proportion p_c of clustered points, the mean cluster size μ_c and the mean cluster radius r_c, in dependence of p_c, μ_c, r_c and the proportion p_{mc} of points in metaclusters. The H-function is only used to estimate r_c. In some plots, the y-axis is splitted for better readability.

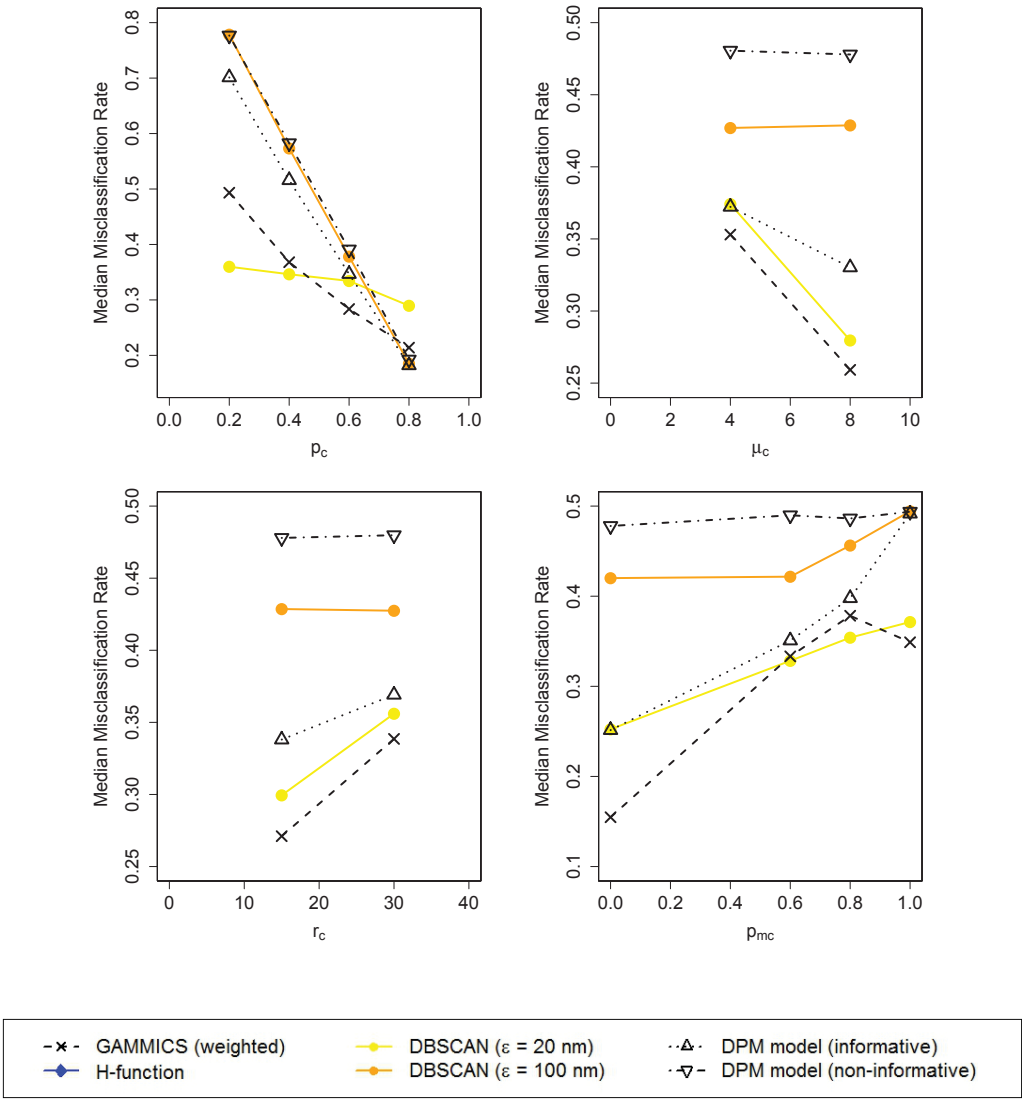

Figure 4.7: Simulation results. Median misclassification rate in dependence of the expected proportion p_c of clustered points, the mean cluster size μ_c, the mean cluster radius r_c and the proportion p_{mc} of points in metaclusters.

a part of the simulation settings in Figures 4.7, 4.8, 4.9 and 4.10. In addition, results for each simulation setting are given in Appendix A.4.

The GAMMICS method performs generally well compared to the other methods, both in terms of estimation accuracy and misclassification rate w.r.t. whether points are clustered or not. DBSCAN is solely able to outperform it in a few cases, but only when an ε value close to the (generally unknown) true

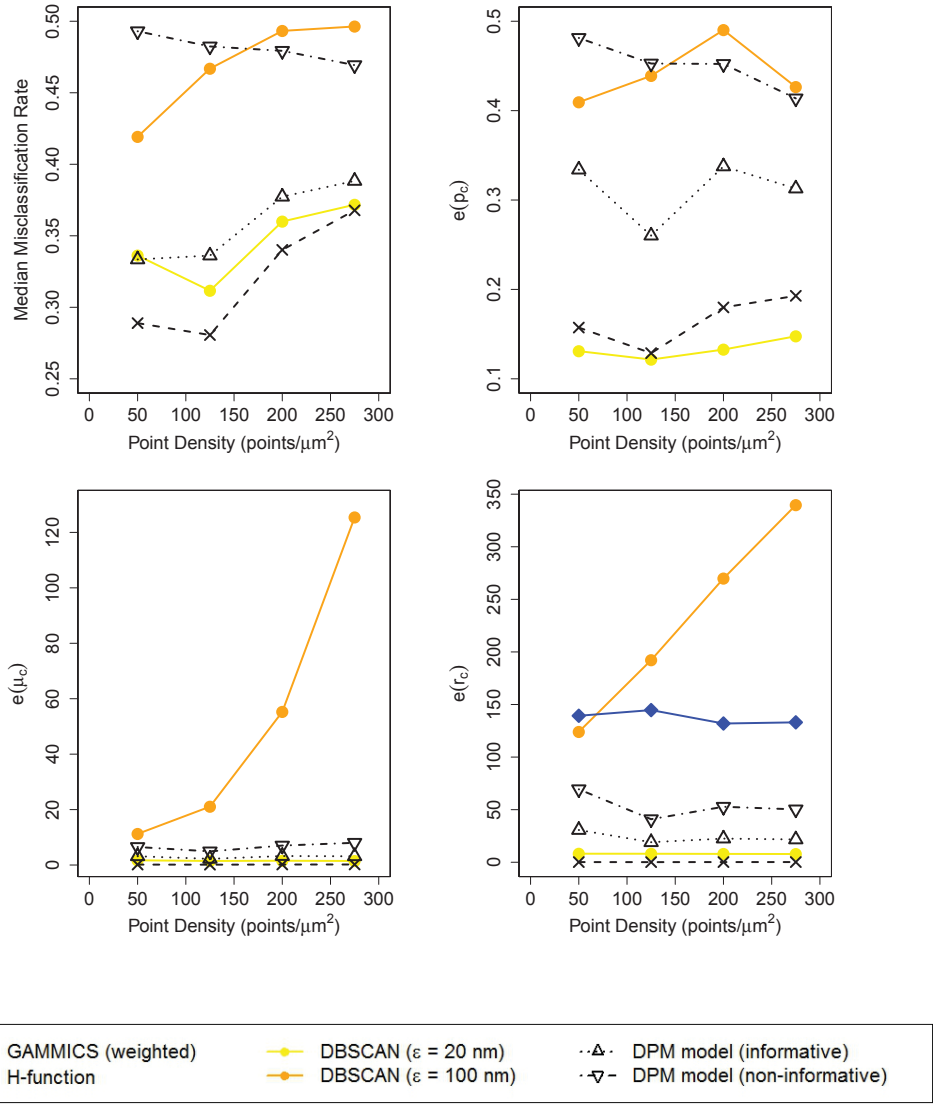

Figure 4.8: Simulation results. Median misclassification rate and median absolute estimation errors $e(\cdot)$ for the expected proportion p_c of clustered points, the mean cluster size μ_c and the mean cluster radius r_c in dependence of the point density (points/μm^2).

value is supplied. Since otherwise the DBSCAN performance decreases enormously, it depends heavily on this parameter choice. Even when ε is chosen close to the true value, GAMMICS often performs better than DBSCAN.

The DPM model performs much better with informative than with non-informative priors. However, even when applied with informative priors, it generally performs worse than GAMMICS and DBSCAN with $\varepsilon = 20$ nm,

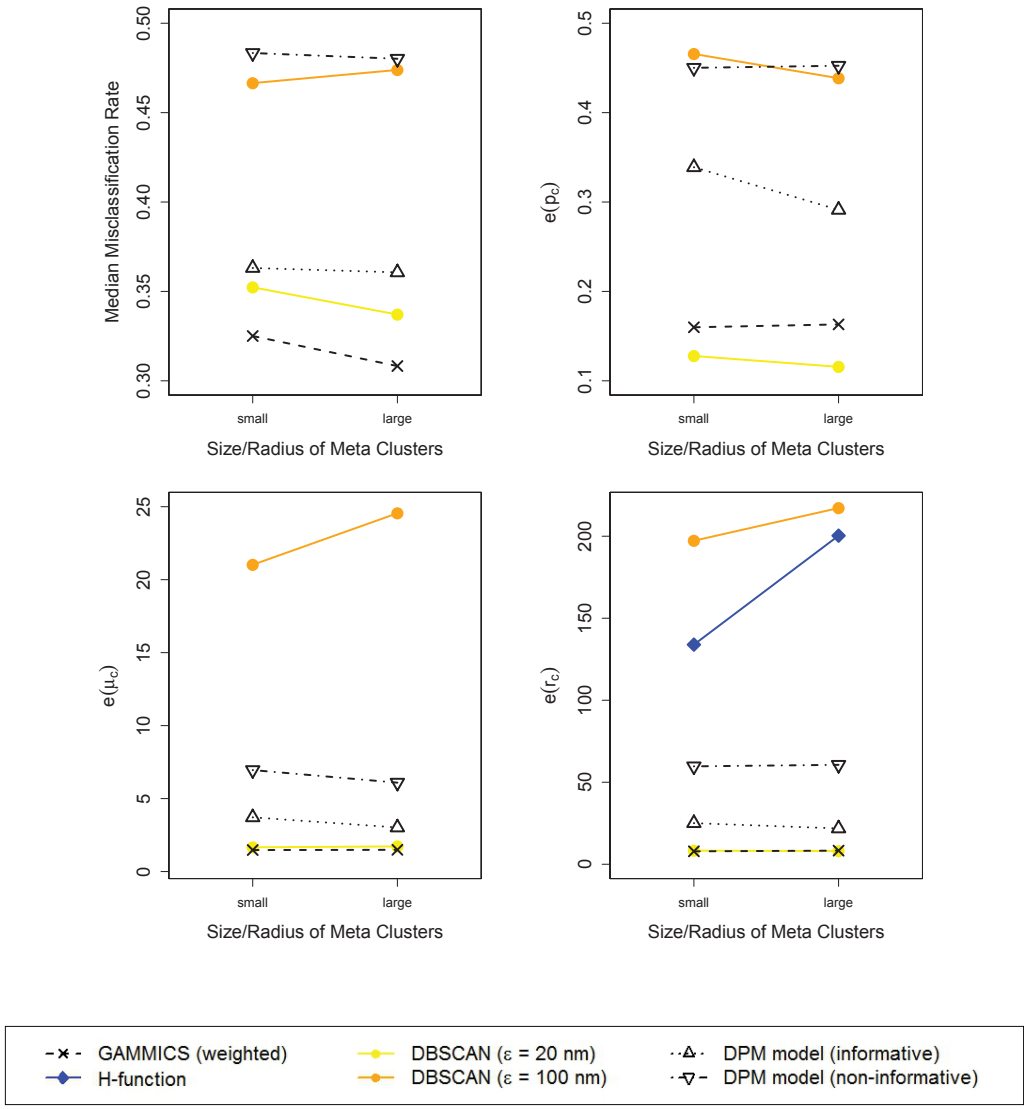

Figure 4.9: Simulation results. Median misclassification rate and median absolute estimation errors $e(\cdot)$ for the expected proportion p_c of clustered points, the mean cluster size μ_c and the mean cluster radius r_c, in dependence of the size and radius of metaclusters, μ_{mc} and r_{mc}. Only the combinations $\mu_{mc} = 20$, $r_{mc} = 100$ nm and $\mu_{mc} = 30$, $r_{mc} = 150$ nm are considered.

except for classification.

Apart from these differences, there are also several aspects the performances of all compared methods have in common. For instance, a bigger cluster size tends to improve the results, while a bigger cluster radius tends to worsen them. A possible explanation is that a bigger cluster size makes the structural

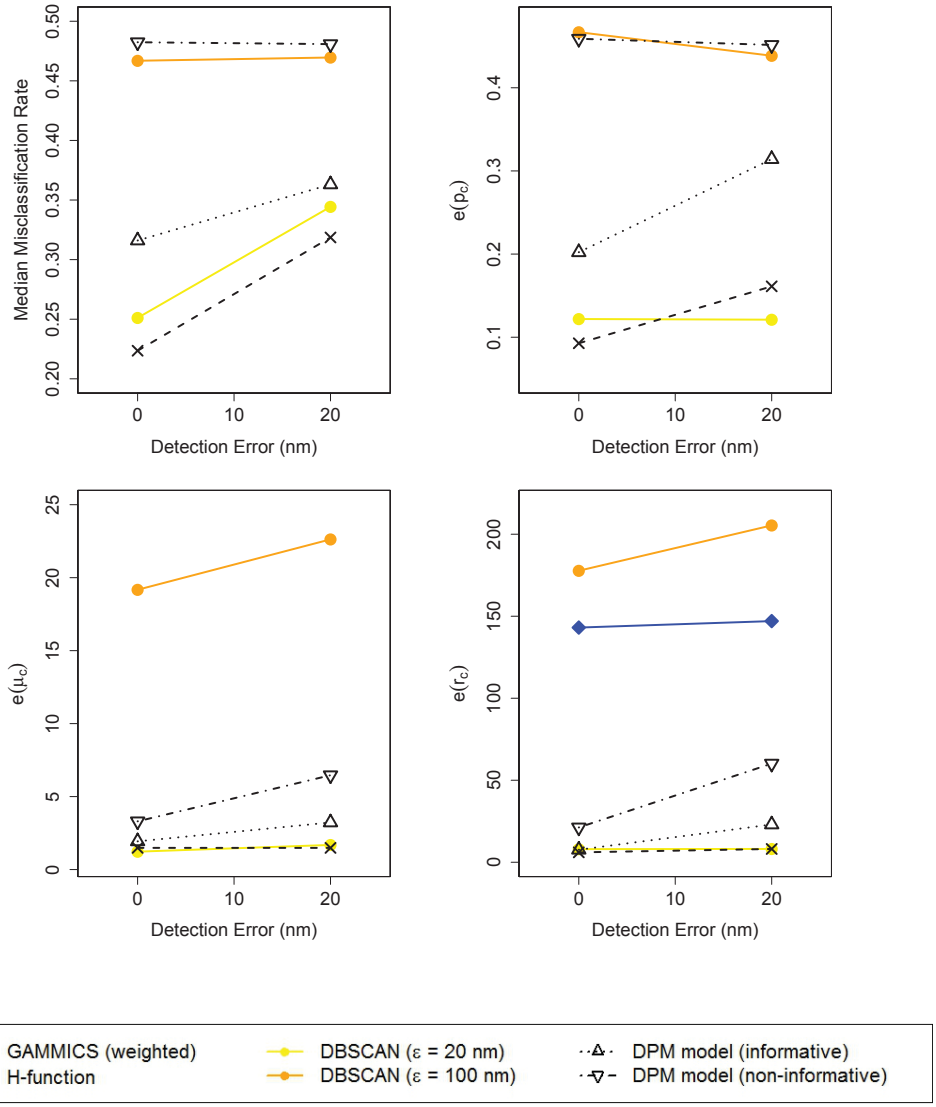

Figure 4.10: Simulation results. Median misclassification rate and median absolute estimation errors $e(\cdot)$ for the expected proportion p_c of clustered points, the mean cluster size μ_c and the mean cluster radius r_c, in dependence of the detection error σ_d.

differences between point patterns consisting of clusters vs. of singletons greater, while a bigger cluster radius makes these differences smaller. All methods tend to perform better when a higher percentage of points is clustered, i.e. when there are more data to learn about the clusters. On the other hand, an increasing occurrence of metaclusters mostly causes all methods to perform worse. Since this effect is often larger for smaller metaclusters which are more similar to

clusters, it might be attributable to confounding of clusters and metaclusters (see Figure 4.9).

In addition to an increasing metaclustering, an increasing point density mostly causes the performance of all methods to deteriorate as well (see Figure 4.8). This is understandable insofar as in a point pattern with a higher density, all distances are smaller, affecting the distinction of singletons, clusters and metaclusters. Naturally, the presence of a detection error worsens results as well (see Figure 4.10).

The classification performance and the estimation accuracy for p_c show similar tendencies in general. This is not surprising: in the GAMMICS model in (4.12), the expected proportion of clustered points p_c is the parameter of the binomial distribution assigned to the allocation variables $T_i, i = 1, \ldots, n$, which eventually are the basis for determining the classification of a point.

Methods with a weak overall performance sometimes manifest departures from the observed common trends. In some cases, many unfavorable circumstances concur in causing the patterns of clustered points and the patterns singletons to be overly similar, e.g. few clustered points in clusters, large cluster radii, small cluster sizes, many and small metaclusters, or a high point density. Then, all methods have misclassification rates in the range of guessing, and estimation accuracy is poor.

The H-function in most cases performs poorly in estimating the cluster radius. While it is competitive when $p_{mc} = 0$, its performance is affected seriously by the presence of metaclustering. This underscores the need for taking the metaclustering into account in the model employed to simulated point patterns.

Several exemplary comparisons of all methods with the Bayesian DBSCAN (reported in the Appendix, Section A.4, Tables A.1 and A.2) indicate that this approach estimates μ_c very well, while its performance when estimating r_c is comparable to the DPM model with non-informative priors. The estimates for p_c tend to be near 1 for all data, perhaps because the method explicitly fixes *MinPts*=1 in the DBSCAN runs. Therefore, they appear to be of little utility.

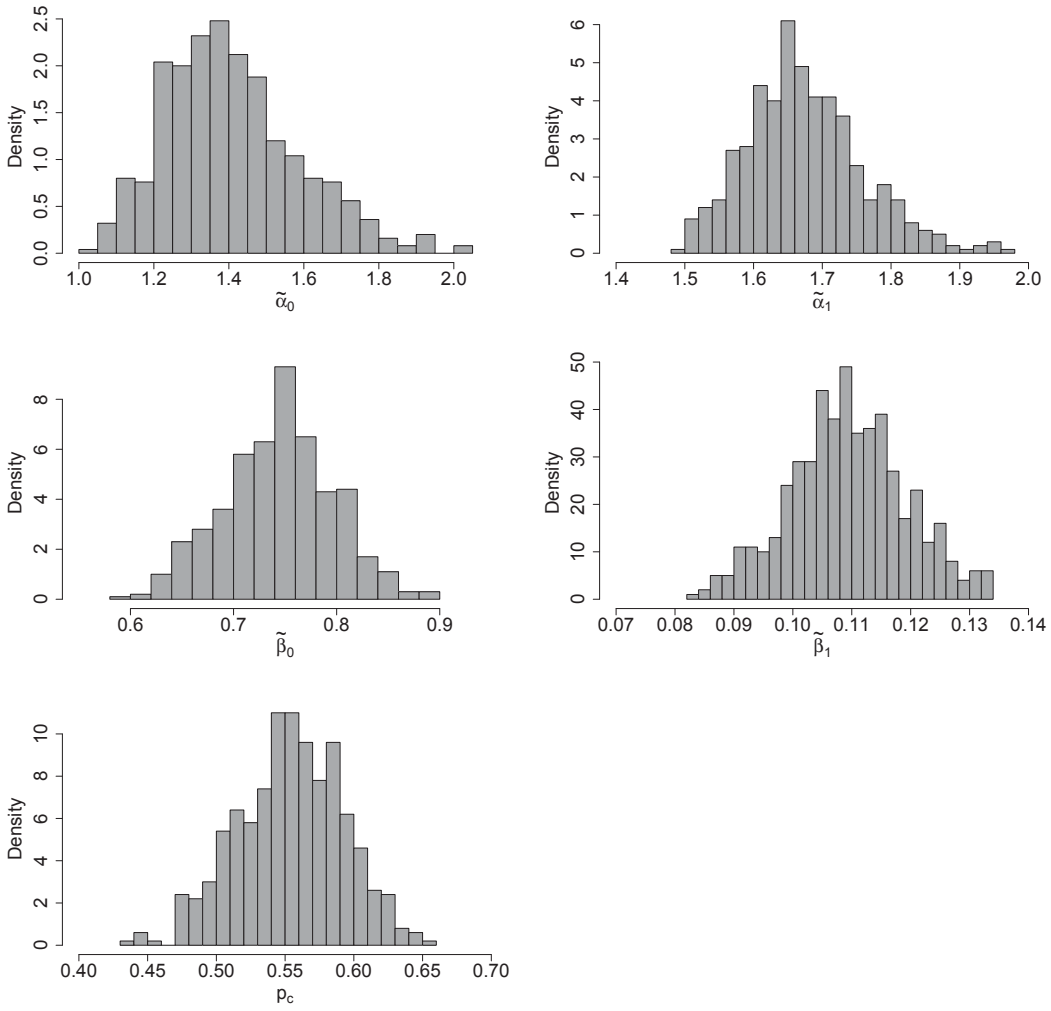

Figure 4.11: Posterior histograms of model parameter distributions across MCMC iterations for localizations simulated with a point density of 200 points/μm^2, proportion of proteins in clusters $p_c = 0.4$, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 15$nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, with a proportion of proteins in metaclusters of $p_{mc} = 0.6$. Shown are histograms for the parameters of the two gamma distributions as well as the proportion of clustered points p_c.

In the GAMMICS method, the uncertainty in the parameters of the gamma distributions and the uncertainty in the binomial allocation distribution are connected as integral model parts. However, these uncertainties are also reflected in the distribution of the cutoff $\tilde{d}_c$ and may propagate into the distributions of the clustering estimates. To get an impression of the propagation of these un-

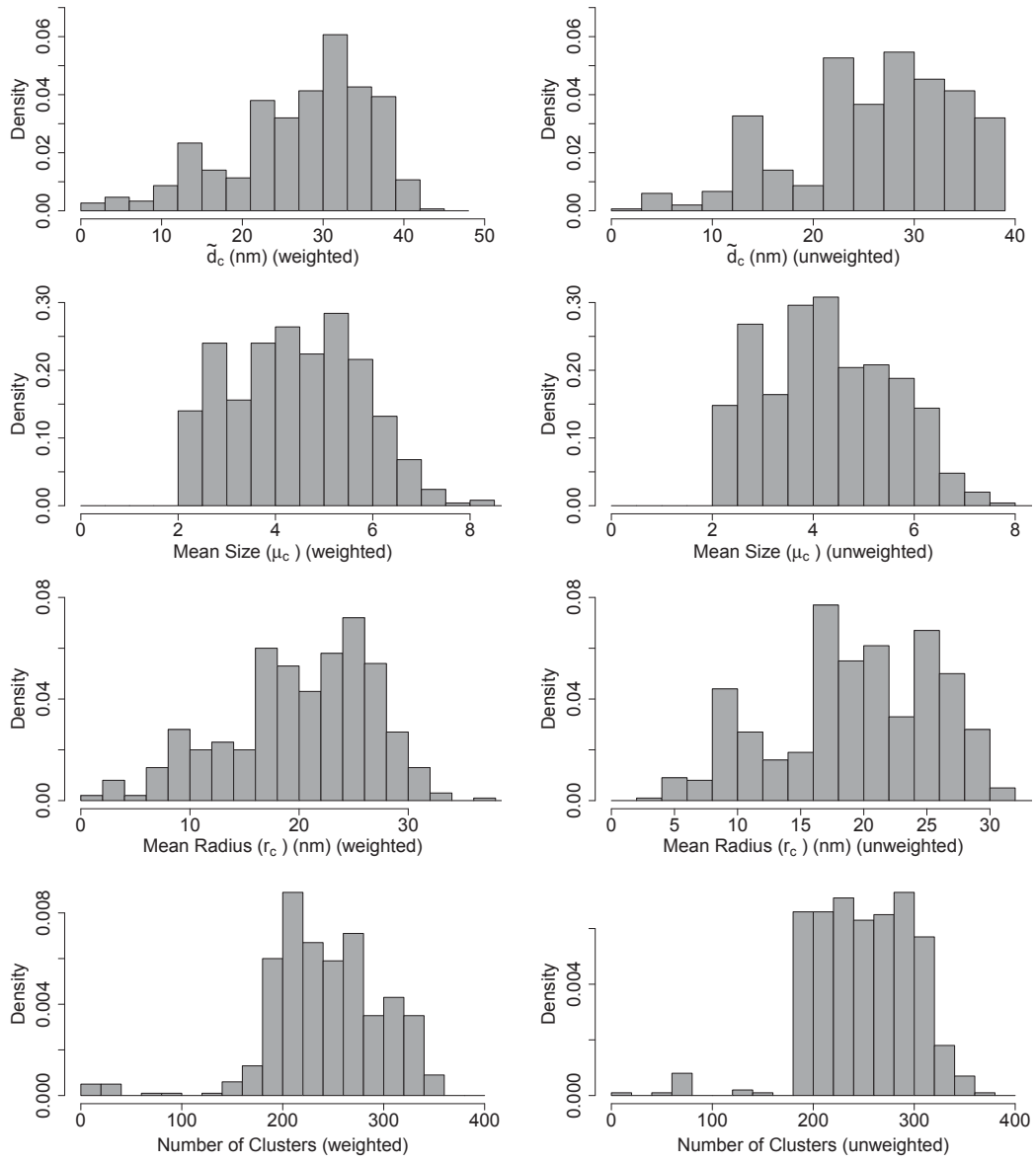

Figure 4.12: Posterior histograms of clustering parameter distributions across MCMC iterations for localizations simulated with a point density of 200 points/μm^2, proportion of proteins in clusters $p_c = 0.4$, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 15$nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, with a proportion of proteins in metaclusters of $p_{mc} = 0.6$. Shown are histograms for the cutoff L_c, the mean cluster size μ_c, the mean cluster radius r_c and the number of clusters. Each histogram is shown twice, on the left hand side for the calculation of L_c with weighted gamma distributions and on the right hand side or the calculation of L_c with unweighted gamma distributions.

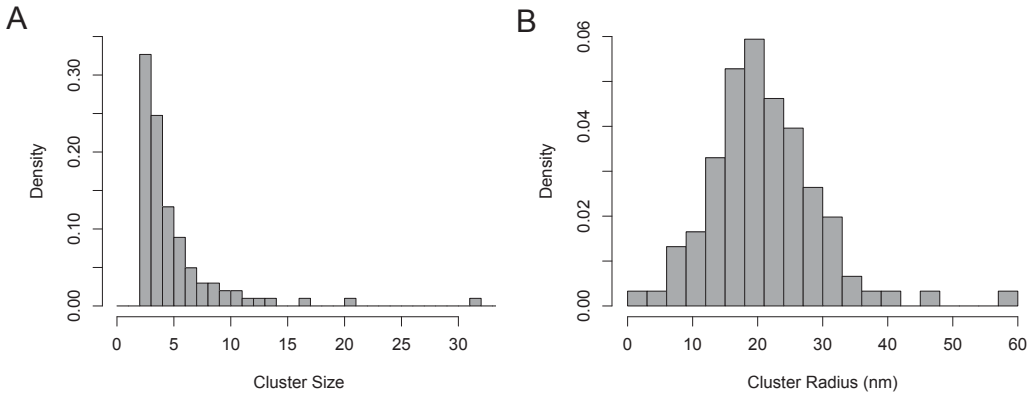

Figure 4.13: Posterior histograms of (**A**) the cluster size and (**B**) the cluster radius for localizations simulated with a point density of 200 points/μm^2, proportion of proteins in clusters $p_c = 0.4$, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 15$nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, with a proportion of proteins in metaclusters of $p_{mc} = 0.6$, based on 100 posterior quantiles of the distribution across the clusters (0%, 1%, 2%, ... quantiles).

certainties, posterior histograms across MCMC iterations are shown for simulated data in Figures 4.11 for the model parameters and in Figure 4.12 for the clustering parameters, readily obtained from the Markov chains in the Bayesian framework (histograms for further data are shown in Figures A.13 and A.14 in Appendix A.4). Distributions of model parameters are mostly unimodal and symmetric, as expected, while the distribution of $\tilde{d}_c$ and the parameters derived from it tend to have more irregular shapes. Unsurprisingly, a left skew in the distributions of $\tilde{d}_c$ often propagates into the distribution of r_c, leading these two distributions to have similar shapes.

Cluster size and cluster radius vary across the different clusters in a cell, but are analyzed here generally in terms of their mean only. Therefore, it is sensible to look at posterior histograms across clusters in addition to the point estimates. In Figure 4.13, such histograms are shown for an exemplary simulation setting. It can be appreciated that they contain considerably more information than just the point estimates. In the histogram for the cluster radius, most of the probability mass concentrates roughly on the range between 10 and 30 nm.

method	mcr	$e(p_c)$	$e(\mu_c)$	$e(r_c)$
GAMMICS (weighted densities)	0.308	0.152	1.539	7.395
GAMMICS (unweighted densities)	0.308	0.152	1.395	7.120
DBSCAN ($\varepsilon = 20$ nm)	0.333	0.130	1.512	8.030
DBSCAN ($\varepsilon = 100$ nm)	0.428	0.422	27.979	215.031
DPM model (informative)	0.356	0.289	2.879	21.144
DPM model (non-informative)	0.478	0.435	6.000	47.217
H-function	-	-	-	139.151

Table 4.3: Aggregated results across all settings of the simulation study for the misclassification rate (mcr) as well as median absolute estimation errors e for the proportion of clustered points p_c, the mean cluster size μ_c and the mean cluster radius r_c.

There are a few radii exceeding this range, possibly arising due to the false classifications of some few singletons as clustered points, increasing the radii of the clusters to which they are assigned. The histogram for the cluster size has a sharp cutoff on the left because singletons are not counted as clusters, making two the smallest possible value. Furthermore, the cluster size distribution is strongly skewed to the right, i.e. there are a number of clusters that contain a high number of points.

The overall aggregated results of the simulation study as presented in Table 4.3 show that GAMMICS performs best w.r.t. classification and estimating r_c, whereas DBSCAN with $\varepsilon = 20$ nm is slightly better in estimating p_c and μ_c. However, if the gamma distributions are not weighted in (4.19) when calculating the cutoff d_c^2, GAMMICS on average performs better than DBSCAN in estimating μ_c. This indicates that this change leads to a convenient robustification even if it is less intuitive. For instance, the change might reduce the impact of a potential bias in the estimation of p_c due to an asymmetric overlap between the two gamma distributions.

DBSCAN, its Bayesian version as well as the DPM model all depend heav-

ily on prior knowledge of the clustering parameters in order to perform well. By contrast, GAMMICS uses informative priors only for the two gamma distributions modeling the distances, which can be often specified with much less knowledge of the cluster structure. Taking into account that prior knowledge of Ras clusters is still not taken for granted, the results confirm GAMMICS' structural advantages.

4.7.2 Experimental Data

In experimental data, the true clustering parameters are unknown, it is thus difficult to quantify the performance of estimation methods. However, methods are developed to work on experimental data in order to answer research questions, and an assessment of results on such data is mandatory. Thus, for further comparison of the methods and for validation, six regions of interest (ROIs) of a cell expressing Ras tagged with the fluorophore mEos2 (see Figure 4.1) are analyzed in this section. The parameter estimations are given in the Appendix, Table A.11, and are visualized in Figure 4.14 in dependence of the point density. This is the only obvious characteristic of the analyzed point patterns that has a potential influence on the methods' performance and does not have to be estimated itself first. The analyzed regions are chosen to cover different point densities, containing 941, 1 039, 1 397, 1 724, 2 139 and 2 677 detected protein localizations in an area of 4 280 x 4 280 nm^2.

The point density appears to have an influence on the estimations, raising the question of whether some methods are susceptible to an unwanted bias arising from differing point densities. For instance, the estimates obtained by DBSCAN (case $\varepsilon = 100$ nm), Bayesian DBSCAN and the DPM model for μ_c and r_c tend to increase with the density. The increase is particularly strong for DBSCAN ($\varepsilon = 100$ nm), suggesting that $\varepsilon = 100$ nm is again not an optimal choice, as already in the simulation study. The estimates obtained by DBSCAN ($\varepsilon = 100$ nm), the DPM model (non-informative case) and Bayesian

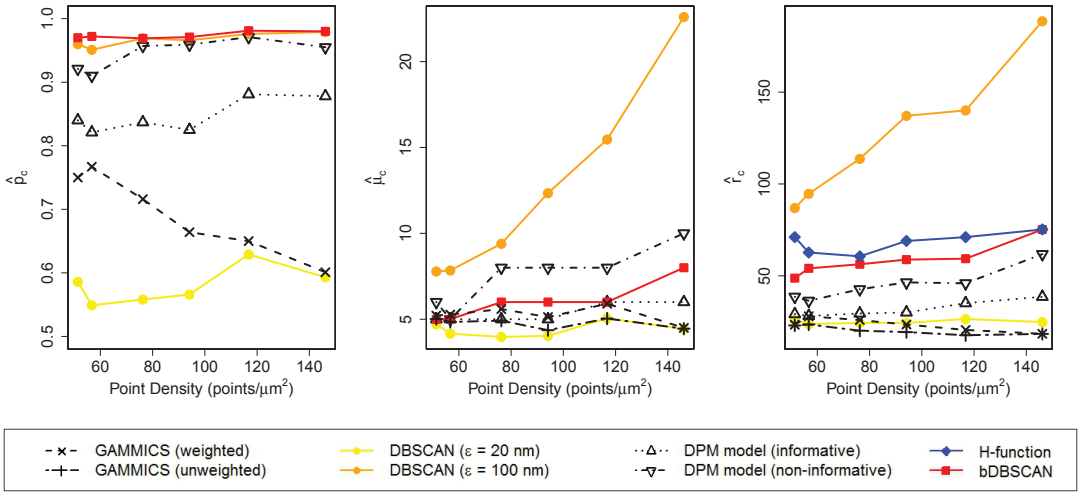

Figure 4.14: Estimates for the proportion of clustered points ($\hat{p}_c$), the mean cluster size ($\hat{\mu}_c$), and the mean cluster radius ($\hat{r}_c$), for regions of interest of the data shown in Figure 4.1, presented in dependence of the point density (points/μm^2).

DBSCAN of p_c are near the maximum value of one and moreover exhibit little variance, raising doubts on the validity of these estimates. For GAMMICS, not weighting the gamma distributions when calculating d_c^2 in (4.19) tends to lead to smaller estimates for μ_p and r_c, which is expected given that $\hat{p}_c > 0.5$. The Bayesian DBSCAN estimates of μ_c are roughly similar to those of GAMMICS and DBSCAN ($\varepsilon = 20$ nm), as could be seen in the simulated data as well, while the Bayesian DBSCAN estimates of r_c are similar to those of the H-function. The GAMMICS estimates of p_c seem to decrease for increasing point densities. Since no point density induced bias is confirmed in the simulation study (see Figure 4.8), one reason might be that some relevant characteristics of the point patterns vary systematically between regions of high and low point density in the analyzed experimental data.

The relative similarity of results obtained by DBSCAN ($\varepsilon = 20$ nm) and GAMMICS in the simulation study can be reported for the experimental data as well. Also, both methods are less affected by the point density than most others. Since, in addition, the results suggest that $\varepsilon = 20$ nm is a better choice

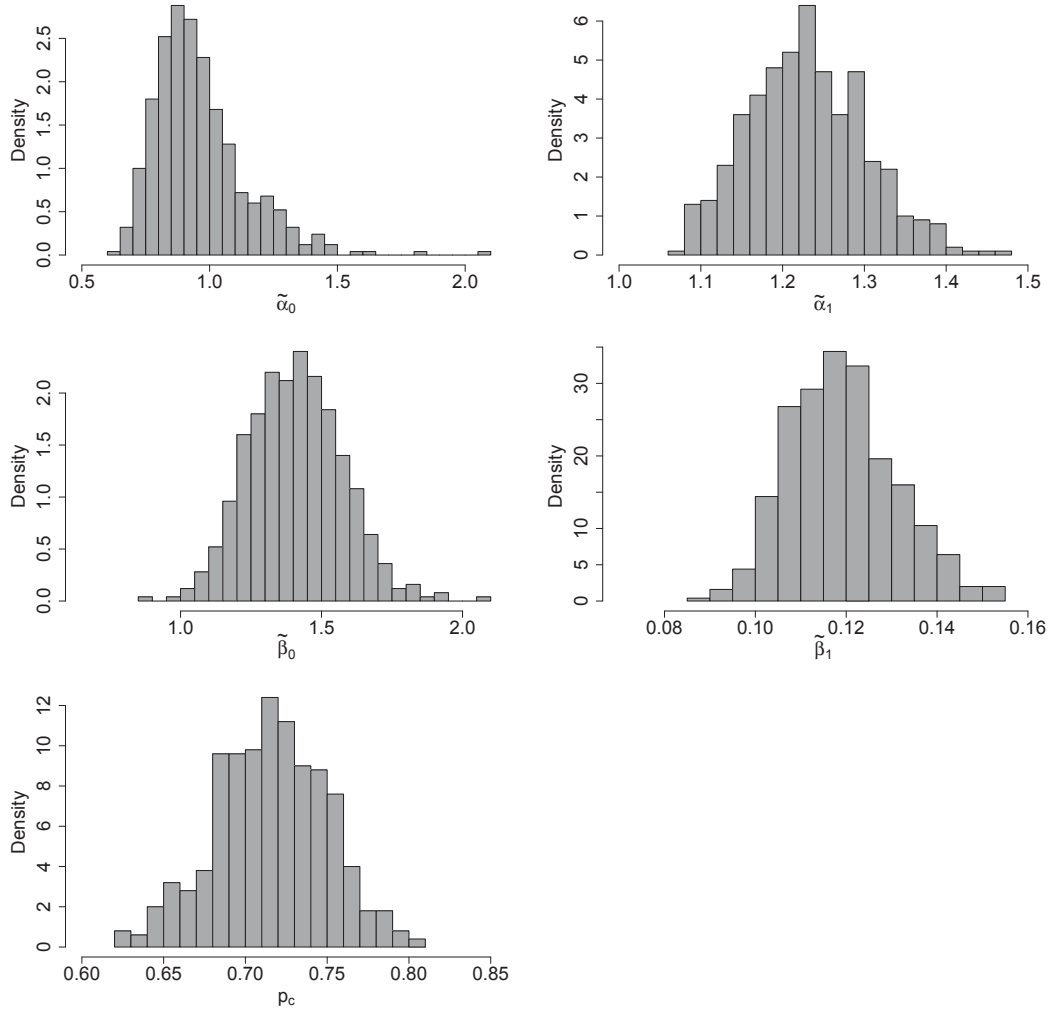

Figure 4.15: Posterior histograms of model parameter distributions across MCMC iterations for experimental data, ROI 3. Shown are histograms for the parameters of the two gamma distributions as well as the proportion of clustered points p_c.

than $\varepsilon = 100$ nm for DBSCAN, it may be suspected that DBSCAN ($\varepsilon = 20$ nm) and GAMMICS present smaller errors for experimental data than the other methods. This would be in congruence with the simulation study. However, as opposed to the simulation study, these results are based on relatively little data and thus have to be interpreted with more caution.

The acceptance rates achieved by GAMMICS in the Metropolis-Hastings step for $\tilde{\alpha}_0$ and $\tilde{\alpha}_1$ are documented in the Appendix, Table A.11. They gener-

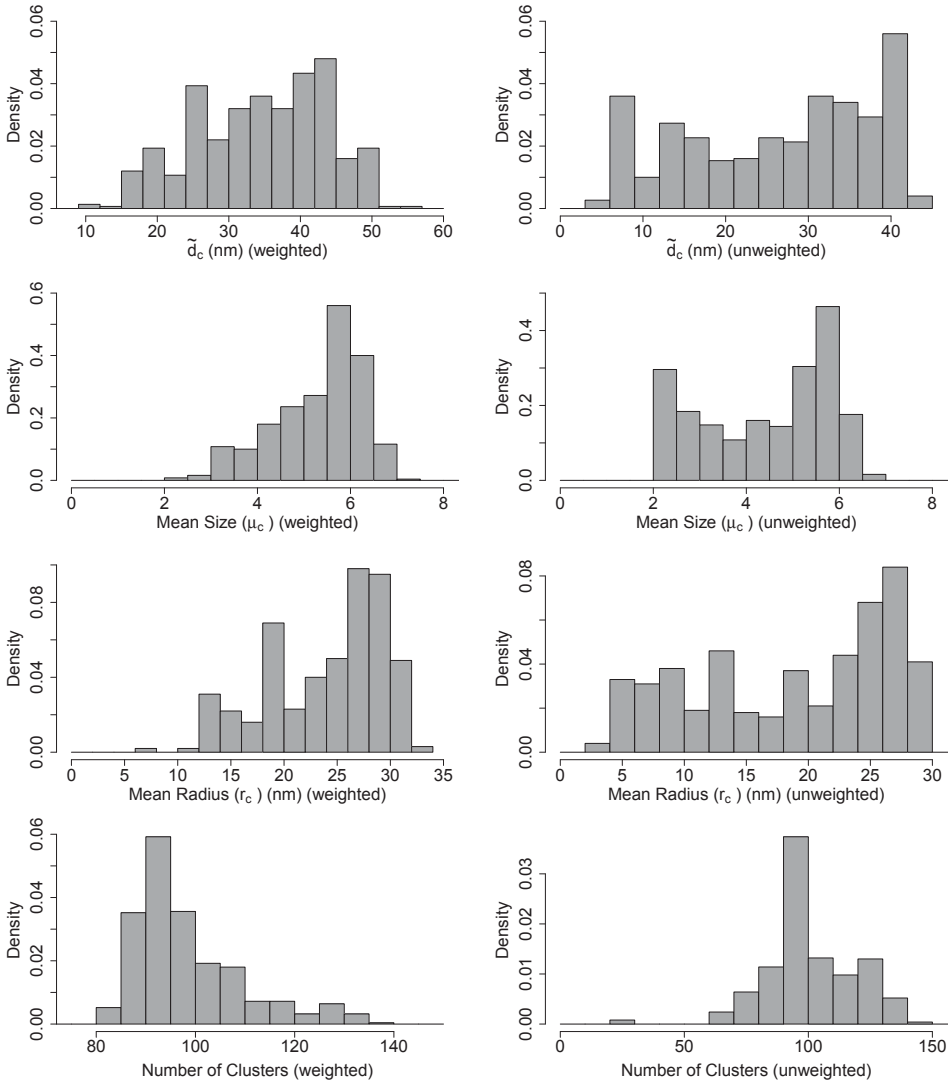

Figure 4.16: Posterior histograms of clustering parameter distributions across MCMC iterations for experimental data, ROI 3. Shown are histograms for the cutoff $\tilde{d}_c$, the mean cluster size μ_c, the mean cluster radius r_c and the number of clusters. Each histogram is shown twice, on the left hand side for the calculation of $\tilde{d}_c$ employing weighted gamma distributions and on the right hand side for the calculation of $\tilde{d}_c$ employing unweighted gamma distributions.

ally decrease with increasing point density, i.e. with potentially more complex shapes of the empirical distribution of distances $d_i^2, i = 1, \ldots, n$.

Posterior histograms produced after fitting GAMMICS to the data from ROI 3 of the experimental data are shown in Figure 4.15 for the model parameters and in Figure 4.16 for the clustering parameters. Again, distributions of model

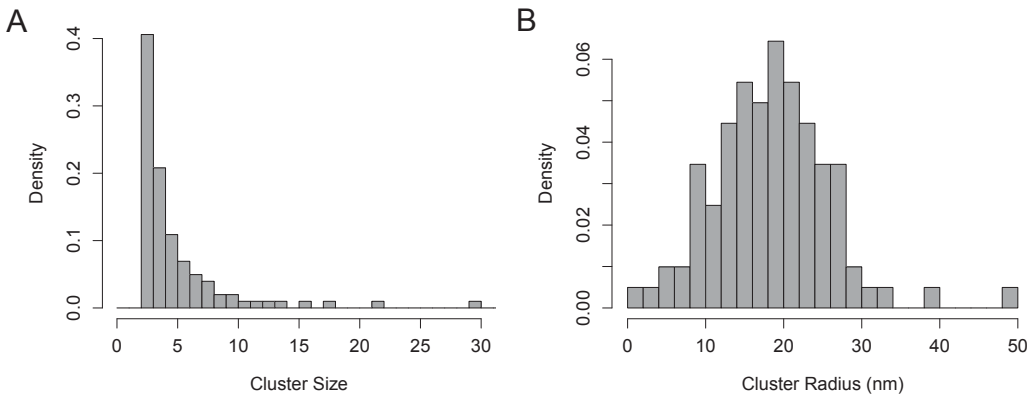

Figure 4.17: Posterior histograms of the cluster size (**A**) and cluster radius (**B**) for experimental data, ROI 6, based on 100 posterior quantiles of the distribution across the clusters (0%, 1%, 2%, ... quantiles).

parameters are mostly unimodal and symmetric, while the distribution of $\tilde{d}_c$ has a more irregular shape, which propagates into the distributions of the clustering parameters. The histograms for $\tilde{d}_c$ and r_c again present a notable left skew. Similar posterior histograms for the data from ROI 6 of the experimental data can be found in Figures A.11 and A.12, Section A.4 of the Appendix.

As in the simulation study, it is reasonable to look at posterior histograms across clusters in addition to the point estimates. Such histograms are presented in Figure 4.17 for the example ROI 6 in the experimental data. They look structurally similar to the histograms obtained from simulated data (Figure 4.13). In the histogram for the cluster radius, most of the probability mass concentrates roughly on the range between 10 and 30 nm. There are a number of radii exceeding this range, however, possibly arising due to some misclassifications of some singletons as clustered points.

Again, the histogram for the cluster size has a sharp cutoff on the left at the minimum cluster size of two. Furthermore, the cluster size distribution has a long and thin right tail, i.e. there are a few clusters which contain many points.

The overall results obtained by GAMMICS on experimental data are roughly in line with previously published results on Ras clustering: based on H-function

Monte Carlo tests simulating possible parameter value over a large grid in a trial and error fashion, so far proportions of 30-45 % of Ras proteins have been reported to form nanoclusters with an average radius of 6-16 nm and an average cluster size of 6-7 proteins (Plowman *et al.*, 2005; Tian *et al.*, 2007; Kiskowski *et al.*, 2009; Shalom-Feuerstein *et al.*, 2008). Although the truth regarding the clustering parameters is not known, the relative similarity observed for distributions across clusters in simulated and experimental data support the validity of the GAMMICS approach when compared with these findings.

Chapter 5

Integrative Analysis of Omics Data

This chapter is dedicated to Bayesian mixture models designed for integrative analyses of sequencing-based epigenetic measurements on histone modifications and another omics data type.

First, a literature review on the analysis of omics data involving sequencing experiments is given with special regard to mixture models and integrative analyses (Section 5.1). Then, it is discussed how the externally centered correlation coefficient (Schäfer *et al.*, 2009) can be adapted for the needs of sequencing data with small sample sizes (Section 5.2). A standard Bayesian mixture model employing a fixed number of Gaussian distributions is devised following Schäfer *et al.* (2012), investigating the degree of consistence in ChIP measurements obtained by microarray and sequencing technology (see Section 5.3). In this model, each class/cluster/group may be represented by several mixture components.

Subsequently, in Section 5.4, a mixture model is developed following Klein *et al.* (2014) that fits a fixed number of components corresponding to different types of distributions. In this model, one class may be represented by several mixture components as well. As will be seen, the use of different types of distributions reduces the number of necessary mixture components and is ben-

eficial for interpretation. While a similar approach using an EM algorithm has been applied to ChIP-seq data by Taslim *et al.* (2009), the innovation presented in Section 5.4 lies in the new input measure and the usage of the Bayesian framework. The latter comes with additional benefits such as the possibility of assessing the uncertainty through credible intervals and less dependence on initial values when applying the method to different data sets. The advantages of the presented method in comparison to a naive two-step analysis are demonstrated as well. The model is employed to identify driver transcripts or probes by looking for genomic loci exhibiting both differential gene transcription and differential histone modification, where these differences have the same direction in both variables (for instance, transcripts identified as both overexpressed and overacetylated would be regarded as potential drivers).

5.1 Methodological Review

In univariate statistical analyses of sequencing data, the existence of differential gene transcription or histone modification is often assessed by assuming a probabilistic model for read counts. In frequentist analyses, this mostly implies the construction of a null distribution for significance tests. The most intuitive distributional assumption for read counts is a Poisson distribution (see, e.g., Jiang and Wong, 2009; Li *et al.*, 2012). Nevertheless, many authors have argued that a negative binomial distribution is more convenient, since its two parameters allow a higher degree of flexibility than the single parameter of the Poisson distribution (see, e.g., Ji *et al.*, 2008, for ChIP-seq data, or, e.g., Anders and Huber, 2010, Di *et al.*, 2011, and Leng *et al.*, 2013, for RNA-seq data). For instance, this has benefits in the case of overdispersion. Further two- or three-parameter generalizations of the Poisson distributions have also been proposed for the analysis of sequencing data (see, e.g., Auer and Doerge, 2011; Esnaola *et al.*, 2013; Srivastava and Chen, 2010). For an even greater flexibility, nonparametric approaches may be considered. Since normalization methods applied to

the read counts after alignment (cf. Section 2.1 and Appendix A.9) may result in continuous data, alternatives to directly modeling read counts are necessary.

Parts of the DNA that play an important role in cancer development may be searched for on a gene-wise basis or by focusing on genomic regions. From a wider perspective, such regions may also be general gene sets enriched for gene transcription or a specific histone modification. In enriched regions, differential gene transcription or histone modification occurs more often than would be expected by chance. In a frequentist context, obtained p-values have to be corrected for multiple testing, e.g., based on the false discovery rate (Benjamini and Hochberg, 1995). In a Bayesian framework, explicit adjustments with regard to multiplicity issues are much less common. Here, specific priors may be used to account for multiplicity (see, e.g., van de Wiel *et al.*, 2012).

A key problem of any analysis involving sequencing data are small sample sizes: due to high costs, to date even a single sample per condition is still not uncommon, largely impeding the modeling of uncertainty across samples on a gene-wise level. When fitting distributions, this poses particular challenges to estimate dispersion-related model parameters. The most popular strategies in this context are essentially to restrict the parameter space for fitting distributions on a gene-wise level and to share information across genes in some way, shrinking one or several model parameters. For the second option, additional information is required to decide between which genes information should be shared. Both strategies have already been pursued frequently in the context of univariate omics analyses. Early and well-known works following this strategies are, e.g., Robinson and Smyth (2007) and Anders and Huber (2010), analyzing RNA-seq data.

Approaches modeling read counts within genomic bins under parameter restrictions, in combination with a search for enriched bins, are related to both ideas. Examples are Kuan *et al.* (2011) or Zeng *et al.* (2013) for ChIP-seq data and Li and Tibshirani (2013) for RNA-seq data.

In particular for RNA-seq data, many methodological approaches have been

proposed. A focus lies on tests for differential transcription. Such methods are mostly based on the Poisson (see, e.g., Li *et al.*, 2012; Marioni *et al.*, 2008) and negative binomial distribution (see, e.g., Robinson and Smyth, 2007; Anders and Huber, 2010; Yu *et al.*, 2013) or generalizations of them (see, e.g., Esnaola *et al.*, 2013). Alternatively, nonparametric tests such as the Wilcoxon rank sum test have been employed (Li and Tibshirani, 2013) as well as tailored tests with specific optimality properties, e.g., maximum average power and control of the FDR (Si and Liu, 2013). Wilcoxon-type test statistics or correlation coefficients have also been employed in permutation tests (Shi *et al.*, 2015).

To generalize beyond a two-group setting, allow for additional covariates, or account for more complex experimental designs such as time-dependent measurements, several methods based on generalized linear models have been proposed. As response variable, they mostly model read counts, assuming the Poisson (Langmead *et al.*, 2010; McCarthy *et al.*, 2012), negative binomial (Hardcastle and Kelly, 2010; Leng *et al.*, 2013), zero-inflated negative binomial (van de Wiel *et al.*, 2012, 2013), or binomial distribution (Zhao *et al.*, 2013). Other responses are modeled as well, e.g., mapping probabilities, assuming a binomial distribution (Zhou *et al.*, 2011). In addition, it has been proposed to transform read counts in order to analyze them with linear regression (Langmead *et al.*, 2010). Law *et al.* (2014) adapt the well-known limma method (Smyth, 2003, 2005) developed for microarray data to RNA-seq data. Some models aim at robustness against outliers (Zhou *et al.*, 2014). Frequentist frameworks generally imply model-based tests, while empirical (Hardcastle and Kelly, 2010; Leng *et al.*, 2013) and fully Bayesian ones (van de Wiel *et al.*, 2012) look to the posterior distribution of model parameters instead.

For comparative reviews of RNA-seq based methods assessing differential gene transcription and a discussion of relevant aspects, see Oshlack and Wakefield (2009), Soneson and Delorenzi (2013), Rapaport *et al.* (2013), or Seyednasrollah *et al.* (2013). Seyednasrollah *et al.* (2015) give a review of corresponding software.

Further methods include enrichment set analyses (e.g., Young *et al.*, 2010) as well as procedures for investigating alternative splicing in RNA-seq data (e.g., Li *et al.*, 2014). Yu *et al.* (2014) transform RNA-seq based gene transcription profiles into documents and apply latent Dirichlet allocation as introduced by Blei *et al.* (2003) to build a topic model.

In ChIP-seq data analysis, many approaches may be applied to investigate either histone modifications or transcription factor binding activity. Since histones usually bind larger stretches of DNA than transcription factors, and moreover different antibodies may tend to bind at regions of different size and different locations within genes, the general applicability of specific methods may be limited. The identification of regions or single loci with a signal clearly distinguishable from noise is referred to as peak finding (cf. Figure 2.5). To determine how many reads are sufficient at a given locus to call it a peak, it is necessary to assess variability in the data. An easy way to call peaks is based on a threshold after calculating a moving average of the reads (Ghosh and Qin, 2010; Pepke *et al.*, 2009). More sophisticated heuristic algorithms include Fejes *et al.* (2008), Hower *et al.* (2011), Wang *et al.* (2013), or Zang *et al.* (2009).

Zhang *et al.* (2008) use a Poisson test to identify differential binding between two conditions. Rozowsky *et al.* (2009) employ an approximate binomial test for the hypothesis that reads in a specific region occur with equal likelihood in samples from both conditions. An algorithm proposed by Kornacker *et al.* (2012) employs chi-square-like test statistics for read counts, while Schweikert *et al.* (2013) advocate a kernel-based non-parametric test.

A method proposed by Mendoza-Parra *et al.* (2013) adapts a regression approach to ChIP-seq data that learns a generative model of multiple binding events in normalized ChIP-chip data (Reiss *et al.*, 2007). The positions of likely binding sites are estimated via a constrained least squares regression. Xu and Zhang (2012) present a negative binomial generalized linear model framework for strand-specific ChIP-seq data, considering a local shifting of peaks.

Cha and Zhou (2014) use Ripley's K-function to detect clustering patterns

in binding sites of two transcription factors.

Mitra and Müller (2014) employ Bayesian hierarchical models - in particular a graphical model and a local bi-clustering approach - to investigate epigenetic associations via ChIP-seq data.

As histone modifications often occur on a scale of hundreds or even thousands of base pairs, the correlation between ChIP signals of adjacent loci (or, e.g., consecutive genomic bins) may be expected to be strong. In this sense, there is certain similarity between ChIP signals and DNA copy number signals. Thus, approaches have been proposed that take such spatial dependence into account. A popular class of methods considering spatial dependence are Hidden Markov or Hidden Ising models, see Ashoor *et al.* (2013), Ernst and Kellis (2010), Qin *et al.* (2010), Song and Smith (2011) and Xu *et al.* (2008) for frequentist examples and Leng *et al.* (2015), Mo (2012) and Spyrou *et al.* (2009) for Bayesian ones. For more complex spatial dependence structures, Markov random fields have been employed. Hu *et al.* (2011), e.g., present a Bayesian Poisson mixed-effects model for the number of RNA-seq reads at the base level. Spatial effects are taken into account via a Markov random field and the overall model is very similar to convolution models known, e.g., in spatial epidemiology to model disease counts (Besag *et al.*, 1991). Another class of methods to take into account spatial dependence are change-point models that estimate the boundaries of segments with a similar average ChIP signal. Herrmann *et al.* (2014) and Xing *et al.* (2012), e.g., present Bayesian approaches pursuing this strategy.

To explicitly analyze and represent interactions between genes, gene regulatory networks may be constructed, either considering a single data input or integrating different data types such as RNA-seq and ChIP-seq. Gene interactions may occur directly when transcription factors bind to their target sequences. More generally, the transcription of a gene may influence the expression of other genes via one or several genes acting intermediately (Wang and Huang, 2014). Most literature on gene regulatory networks until data is based

on microarray data. In recent years, often the pair-wise correlation between genes is calculated as base input. For this purpose, Pearson's or Spearman's correlation coefficient (see, e.g., Giorgi *et al.*, 2013) or an alternative measure such as mutual information or partial correlation is employed.

Other works such as Qin *et al.* (2014) employ penalized regression approaches to infer gene regulatory networks. Graphical Gaussian models assume a joint multivariate normal distribution for all variables (see, e.g., Guan *et al.*, 2014b). Some works also integrate evidence reported in previous publications, such as Gene Ontology pathway information (e.g., Zhang and Mallick, 2013).

The Bayesian framework is particularly suited for expressing causal relationships, combining domain knowledge and data, avoiding overfitting, and for constructing gene networks based on incomplete data sets (Needham *et al.*, 2006). Yu *et al.* (2008) and Guan *et al.* (2014a), e.g., propose a Bayesian network for the integration of ChIP-seq data and general gene transcription profiles. Tang *et al.* (2012) describe a similar framework for time-dependent data. A network presented by Liu *et al.* (2013) integrates ChIP-seq, RNA-seq, and potentially further sequencing data. Gong *et al.* (2015) develop a network for such data when the measurements are time-dependent. To take into account the inherent correlation between neighboring loci in ChIP data, Bayesian networks may offer some advantages over other mentioned approaches. Hoffman *et al.* (2012), e.g., argue that a dynamic Bayesian network can represent a standard hidden Markov model (HMM), representing the HMM's hidden state with a hidden random variable and the observations with an observed random variable. These authors compare a Bayesian network for ChIP-seq and further sequencing data to the structurally similar HMM of Ernst and Kellis (2010), concluding that the Bayesian network is more flexible and appropriate in modeling complex relationships among variables.

An overview of methodological approaches to construct gene regulatory networks is given in, e.g., Emmert-Streib *et al.* (2012) and Lopez-Kleine *et al.* (2013), with a focus on microarray data but also considering sequencing exper-

iments. Bolouri (2014) discusses specific challenges for this model class posed by sequencing data.

Besides gene regulatory networks, further approaches tackle the integrative analysis of distinct omics variables based on sequencing data. Recently, in particular joint analyses of RNA-seq and ChIP-seq data have been published. Shamimuzzaman and Vodkin (2013), e.g., simply analyze both inputs separately by standard differential expression and peak finding approaches and jointly interpret results from both analyses. Wu and Ji (2013) implicitly integrate RNA-seq and ChIP-seq data by correlating transcription factors and their known functional target genes via their transcription levels. They use the correlation of these levels to update the ranking of transcription factor bound genes identified in ChIP-Seq (or ChIP-chip) experiments. In a two-step procedure, Xie *et al.* (2011) first assess abnormalities in RNA-seq data and subsequently check for differential DNA methylation between conditions using a t-test.

Finally, several approaches based on (log) linear regression have been proposed, regressing gene transcription on ChIP-seq based predictors. Cheng *et al.* (2011), Cheng *et al.* (2012), and Dong *et al.* (2012) consider histone modifications or transcription factor activity as regressors (ChIP-seq and ChIP-chip data). They use classification methods such as support vector machines or random forests to classify transcripts regarding their gene transcription levels based on the epigenetic data. Karlić *et al.* (2010) regress gene transcription on the histone modification levels at the genes' promoters. Ouyang *et al.* (2009) consider principal components of selected ChIP-seq transcription factor binding data as regressor variables. They also build CART trees based on the principal components to distinguish gene sets. McLeay *et al.* (2012) propose a similar approach, building joint principal components for transcription factor data, histone modification data and chromatin accessibility scores. Park and Nakai (2011) regress gene transcription values on both transcription factor binding scores and discrete values representing histone modifications. Xu *et al.* (2010) apply a stepwise regression procedure in which potential predictors for gene

transcription values are different histone modification measurements as well as their two- and three way interactions. They also employ a multivariate adaptive regression splines (MARS, see Friedman, 1991) model to capture potential non-linearity in the relation between histone modifications and gene transcription due to saturation effects.

Recent methods integrating RNA-seq and ChIP-seq data are reviewed in Angelini and Costa (2014). More general reviews on integrative omics analyses can be found in Hawkins *et al.* (2010) and Ritchie *et al.* (2015).

Mixture models, the methodological focus in this thesis, have already been used for various types of analyses involving genetic and epigenetic data, including sequencing data. In univariate analyses, a typical research question tackled by mixture models is the clustering or classification of genes. Frequent goals are, in particular, the detection of copy number changes and differential gene transcription, as well as the identification of binding sites for the same transcription factor. Klambauer *et al.* (2012), e.g., employ a mixture of Poisson distributions to model the copy number based on sequencing experiments. Each of the distributions represents one integer copy number and a Dirichlet distribution is a priori assumed for the mixture weights.

Several frequentist mixture models have been proposed for the analysis of RNA-seq gene transcription data. Wang *et al.* (2014), e.g., describe a model involving a mixture of bivariate Poisson distributions to cluster transcripts, where the two dimensions correspond to two biological conditions of interest. Rau *et al.* (2011) present a model based on a finite mixture of Poisson distributions to cluster transcription profiles across several replicates and conditions. Ye *et al.* (2015) fit a mixture of multivariate Poisson distributions (Karlis and Meligkotsidou, 2007) to cluster time series of read counts, assuming an AR(1) model over time for the parameters of the Poisson distributions. In a Bayesian framework, Leng *et al.* (2013), e.g., propose an empirical Bayesian two-component mixture to model RNA-seq data for two conditions of interest. Chung *et al.* (2013) fit a Bayesian hierarchical model assigning a Poisson distribution for

read counts, where the mean is represented as the product of the individual sequencing depth, a gamma-distributed baseline transcription measure and a transcription level fold change induced by a treatment. The fold change is modeled by a mixture of two log-normal distributions, one for increased and one for reduced gene transcription. Guindani *et al.* (2014) develop a semiparametric mixture of Poisson distributions to estimate the distribution of observed sequence counts when overdispersion is present. They assume a Dirichlet process prior on the mean of the Poisson components. This method addresses the analysis of SAGE (serial analysis of gene expression) data, an older sequence-based technology working with small tags corresponding to fragments associated to transcripts.

Many more Bayesian mixture models have previously been proposed to analyze transcription data from gene expression microarrays and other data sources for gene transcription. For instance, Do *et al.* (2005) model gene transcription values with a mixture of two components corresponding to genes exhibiting vs. not exhibiting differential transcription. Each of the two distributions representing the components is in turn represented by a nonparametric mixture of normals assigned a Dirichlet process prior. Dahl (2006), Medvedovic and Sivaganesan (2002), and Rasmussen *et al.* (2009) use nonparametric Bayesian mixture models to cluster genes w.r.t. similar transcription levels. Kim *et al.* (2006) use such models to detect existing subclasses of observations, while identifying genes that discriminate between them in clustering. Joshi *et al.* (2008) propose methods for Bayesian bi-clustering, deriving fuzzy clusters from the posterior similarity matrix based on eigenvectors. Both Wang and Wang (2013) and Liu *et al.* (2006) describe model-based clustering frameworks with an infinite mixture of multivariate Gaussian distributions and an hierarchical Dirichlet process as mixture prior. Newton *et al.* (2004) detect differential gene transcription with an hierarchical model based on a mixture of gamma distributions.

Mixture models have also been applied to ChIP-seq data. Ibrahim *et al.* (2015), e.g., describe a mixture of multivariate Gaussian distributions to cluster

read counts for peak calling. Rashid *et al.* (2011) employ a mixture of three generalized linear models consisting of one zero-inflated component, one background component and one enrichment component, assuming negative binomial distributions for the read count responses. Xu *et al.* (2011) propose a mixture model with nine components to analyze the joint distribution of two motifs within a sequence. They consider each motif to be either absent, present on the plus strand or present on the reverse complementary strand. Gaussian and uniform distributions are considered for the components. Pique-Regi *et al.* (2011) and Kuan *et al.* (2011) fit a mixture of two negative binomial components to ChIP-seq data measured for binding locations or in genomic bins, respectively. Zeng *et al.* (2013) present a more general version of such a model, explicitly considering the case of several available samples. Zhang *et al.* (2011) model local concentrations of ChIP-seq reads with a Bayesian hierarchical mixture of t-distributions in order to identify regions of transcription factor binding. Xu *et al.* (2013) employ a Pólya urn scheme to construct a nonparametric Bayesian model for two-dimensional clustering of histone modifications and genomic locations. Taslim *et al.* (2012) fit a finite mixture of double exponential and uniform distributions by minimizing the Kullback-Leibler distance, analyzing histone modifications. Wei *et al.* (2012) use a finite mixture model with a Dirichlet prior to cluster SNPs with a similar allelic (im)balance (i.e. a similar protein-DNA binding activity in both alleles) across several ChIP-seq data sets.

In a frequentist setting, Taslim *et al.* (2009) fit a mixture of a fixed number of normal and exponential distributions to normalized differences of RNA polymerase II binding activity between conditions of interest.

In integrative analyses of omics data from different sources, mixture models have also been employed, although until date almost only for microarray data and other data types which are older than sequencing experiments. Most of the methods are cluster approaches designed for specific data types. In many of these applications, gene transcription is analyzed jointly with another input such as DNA copy number or transcription factor binding.

Kormaksson *et al.* (2012), e.g., propose a two-component Gaussian mixture model to cluster probe sets according to their methylation state, considering high and low methylation. They extend the model to integrate further continuous variables and/or platforms by introducing cluster and platform specific activity indicators, assuming that objects in a given cluster have identical activity profiles in each data source. Liu *et al.* (2007) present a method based on a hierarchical DPM cluster model for gene transcription by Liu *et al.* (2006). They further incorporate ChIP-chip data, making distributional assumptions for transcription factor binding profiles.

Savage *et al.* (2010) propose a hierarchical DPM model for gene transcription and transcription factor binding as well. The genes are assigned to three contexts: one containing the genes fused across the two data sources via a product of likelihoods, one using only transcription data and the last one using transcription factor binding data. Each context defines a clustering partition via a DPM model. Kirk *et al.* (2012) present a method fitting a mixture model for clustering with a finite dimensional Dirichlet prior to each of several data sources. The models for the different data sources are linked via a joint prior for their component allocations. The method considers to use different types of distributions for different data types, e.g. Gaussian distributions for continuous variables, Gaussian processes for gene transcription time-course data or multinomial distributions for discretized data. Both Savage *et al.* (2010) and Kirk *et al.* (2012) select a single cluster partition via a criterion based on the posterior similarity matrix calculated from the posterior distribution of cluster allocations (Fritsch and Ickstadt, 2009).

In van Wieringen and van de Wiel (2009), a mixture-like model with three components is described for gene transcription, where the mixture weights correspond to probabilities for DNA copy number calls. From this model, they derive weighted versions of nonparametric tests such as the Wilcoxon test. Joint analyses can also be employed to integrate different technologies instead of different data types, which will be the subject of Section 5.3.

Due to the usually small number of clusters or classes of interest in the mentioned specific research fields, the number of clusters is fixed in most specialized methods, while in general methods such as the one of Kirk *et al.* (2012) it is rather not.

In frequentist frameworks, the EM algorithm is usually employed to fit mixture models (as, e.g., in Newton *et al.*, 2004; Taslim *et al.*, 2009; van Wieringen and van de Wiel, 2009). However, the algorithm is also employed in the Bayesian context (in, e.g., Holmes *et al.*, 2012; Leng *et al.*, 2013; Kuan *et al.*, 2011; Zhang *et al.*, 2011). Nevertheless, MCMC algorithms are probably the most common technique employed to fit Bayesian mixture models to omics data.

Input measures for mixture models are highly application-specific. Examples include different log-ratio intensity measurements or transformed p-values. In this thesis, the employed measures are inspired by the externally centered correlation coefficient (Schäfer *et al.*, 2009, described in Section 5.2).

5.2 Adaptation of the Externally Centered Correlation Coefficient

In an integrative analysis of two omics data types, the absolute values from both variables are generally not the most informative measures in medical and biological terms. Differences between values from two collectives representing different biological conditions, e.g., cancer patients and a reference collective, are usually more meaningful when investigating causes of a disease. For similar reasons, reference collectives are important in other statistical research on diseases as well, such as in case-control studies in epidemiology. In omics analyses, the importance of assessing results in relation to a reference collective primarily stems from the fact that in sequencing experiments like ChIP-seq, some DNA fragments cannot be uniquely aligned to the genome ('mappability') and DNA fragments with high guanine-cytosine content are sequenced more efficiently

than others. Such effects give rise to artifacts that can be canceled out by focusing on quotients of values from two collectives instead of values from just one collective (Vega *et al.*, 2009). As a consequence, the use of a reference collective has been the standard for years in the analysis of omics data, and methods not relying on them often take other steps to control mappability and guanine-cytosine content (see, e.g., Zeng *et al.*, 2013). DNA copy number values, e.g., are generally reported as (often logarithmized) quotients between values from two collectives, i.e. a reference collective is commonly used already at low-level analysis stages. For gene transcription and ChIP analyses, on the other hand, a reference collective is often employed at later analysis stages when investigating, e.g., differential gene transcription and differential histone modification regarding different biological conditions.

Approaches that explicitly discuss the importance of assessing a collective of interest relative to a control collective in the context of sequencing experiments are, e.g., Rozowsky *et al.* (2009) (ChIP-seq data) and Chung *et al.* (2013) (RNA-seq data).

For the integrative analysis of several omics inputs, in Schäfer *et al.* (2009) an externally centered correlation coefficient was developed as a means to analyze two input random variables measured on two different collectives in an integrative way. It is defined as follows:

Definition 5.1 (**Externally centered correlation coefficient**)**.**
The externally centered correlation coefficient is defined as

$$r_{EC} := \frac{\sum_{i=1}^{n}(X_i^* - A^*)(Y_i^* - B^*)}{\sqrt{\sum_{i=1}^{n}(X_i^* - A^*)^2} \cdot \sqrt{\sum_{i=1}^{n}(Y_i^* - B^*)^2}} \tag{5.1}$$

for two random vectors $X^* = (X_1^*, \ldots, X_n^*)'$ *and* $Y^* = (Y_1^*, \ldots, Y_n^*)'$*, corresponding to two sets of measurements on* n *subjects, as well as median values* A^* *and* B^* *across samples of two corresponding reference collectives, respectively.*

Due to the application which motivated its invention in Schäfer *et al.* (2009) (X^* represented gene transcription and Y^* copy number measurements for a specific locus on the genome), all measurements are on the $\log_2$-scale.

The coefficient assesses how much a target collective and a control collective differ consistently from either other in the values of the two data sources at hand. The central idea to accomplish this is to replace the two mean values in the definition of a Pearson correlation coefficient by the median values from the reference population(s). Since the reference collective only enters via the two median values, it is possible to use a separate reference collective for each variable. This facilitates the analysis in situations when no control collective is available for which both variables were measured.

The externally centered correlation coefficient has become an established method for the integrated analysis of gene transcription and copy number data and has been shown to perform favorably or comparably to several other state-of-the-art methods (see, e.g., Lahti *et al.*, 2013; Louhimo *et al.*, 2012). As explained in Schäfer *et al.* (2009), the coefficient uses more information than both two-step approaches and correlation or regression approaches. Two-step methods first assess the differences between conditions separately for each input and afterwards search for overlaps between regions identified by each input. Normally, this separate identification of interesting or significant regions involves some kind of dichotomization that loses information. Correlation or regression approaches quantifying the linear relationship between two variables or conditions, on the other hand, fail to simultaneously compare two variables and conditions. In Figure 5.1, it is illustrated that regression (and, thus, correlation) is unable to measure the degree to which equally directed aberrations from the reference collective are present in the cancer collective for both variables. In this section, an adaptation of this framework to the integrative analysis of ChIP-seq data and another input is developed.

One of the problems of current sequencing studies is that the number of samples is very small. One sample per collective is not an uncommon number,

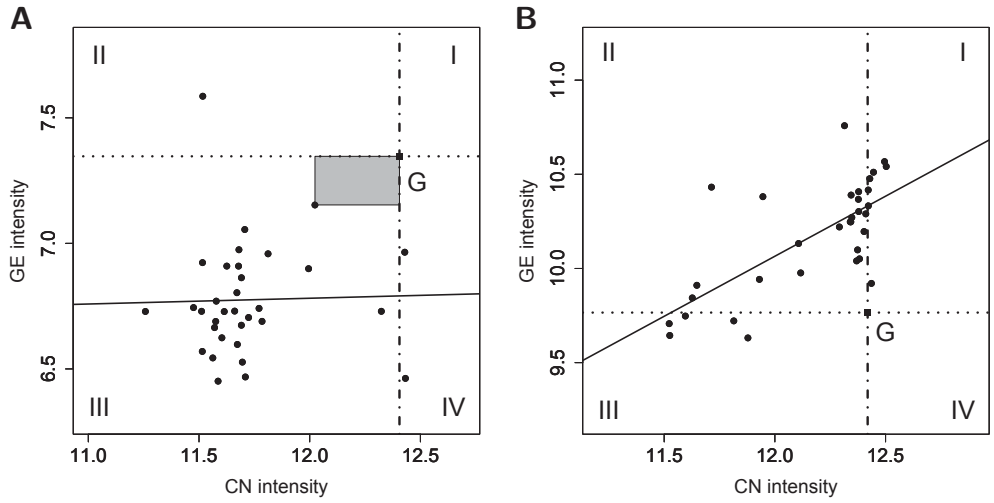

Figure 5.1: The idea of the externally centered correlation coefficient. Copy number (CN) and gene transcription (GE) intensities of 33 patients of acute myeloid leukemia, plotted for (**A**) the 224963_at probe set and (**B**) the 54970_at probe set from the Affymetrix U133 chip set. The dotted and dot-dashed lines represent the median CN and GE intensities in the reference data set, respectively. G is their intersection. The solid line represents the line of a standard linear regression of GE intensity on CN intensity. The gray rectangle measures the deviation of CN and GE from the median of the references for a single patient. The slope of the regression lines does not represent the degree of equally directed abnormalities in both variables. Source: Schäfer *et al.* (2009).

and more than five samples per collective are often not available until date. No type of correlation can be sensibly calculated in such situations. This, in consequence, also applies to r_{EC}. However, the rectangles defined for every single sample by the summands in the nominator of (5.1), used by Schäfer *et al.* (2009) to construct a Wilcoxon rank sum test and illustrated in Figure 5.1A, remain interpretable, and their area can still be employed as a basic measure for analysis.

In the following, the concept of externally centered correlation is adapted to small sample sizes. Two proposals for appropriate measures are made, motivated by two different applications discussed in the remainder of this chapter. Bayesian concepts are convenient in case of such small sample sizes, since – in absence of variation across samples or other additional information – uncertainty can be quantified only across genes. Bayesian methods are an appro-

priate framework to do this, since they can easily incorporate prior knowledge or other additional information to compensate for a small sample size. Conceptually, they focus on distributions as means of measuring uncertainty, while frequentist methods focus on measurement repetitions instead.

In developing a new coefficient, the need for normalization of the rectangle areas arises, so that the values of the coefficient have a comparable range across different data sets. This is important for interpretability and for making distributional assumptions. The measure of dispersion necessary for normalization cannot be calculated across samples if, e.g., there is only one sample. Instead, it has to be calculated across genes or loci.

The normalization approach should account for the fact that genomic regions may have different levels of natural variability in a given variable. This may be due to, e.g., natural groupings of genes occurring in specific genomic variations. Eventually, such variability can not be distinguished from the variability of interest, i.e. the variability due to differences in the two data sources between a target collective and a reference collective. Such unwanted effects can be avoided by performing the normalization within windows of fixed size around each locus i, containing loci $i_l = i - n_{i,l}, i - n_{i,l} + 1, \ldots, i - 1, i, i + 1, \ldots, i + n_{i,r} - 1, i + n_{i,r}$, where the total window contains $n_{i,l} + n_{i,r}$ points. The window width can, e.g., be chosen according to the width of genetic groupings, if available. Since the variables' empirical distributions within the windows – in particular the number of loci per window – may vary considerably, it is advisable to use a robust measure with little sensitivity to such variations. A modified measure for the loci $i = 1, \ldots, n$, inspired by the externally centered correlation coefficient, can then be defined as

$$Z_i := \frac{(X_i^* - A_i^*)}{med(X_{i_l}^* - A_{i_l}^*)} \cdot \frac{(Y_i^* - B_i^*)}{med(Y_{i_l}^* - B_{i_l}^*)}, \tag{5.2}$$

(see Schäfer *et al.*, 2012), where X_i^* and Y_i^* are the random variables for the target sample at locus i, while A_i^* and B_i^* are the corresponding random vari-

ables for the corresponding locus i in the reference sample, and $med()$ is the median. If there is more than one sample, X_i^*, Y_i^*, A_i^* and B_i^* can be calculated as mean or median values across the samples. The $Z_i, i = 1, \ldots, n$, are considered to be identically distributed. As in Schäfer *et al.* (2009), when equally directed differences between the target and the reference cell line are present in both inputs, the corresponding Z_i value is positive, while in case of unequally directed differences it is negative. If a gene shows differences in only one of both data types or has no differences at all, its Z_i value is expected to be equal or close to zero.

Sometimes there exist no genomic groupings potentially leading to blocks of specific natural variability across genes in the variables under consideration. In such cases, the choice of a window in which to perform normalization becomes less obvious and more arbitrary. To facilitate analysis on different data sets, it may then appear more reasonable to rescale all rectangle areas with a factor representing the overall variability across the genome. A correlation value Z_i analogous to (5.2) can then be calculated for each transcript as

$$Z_i := \frac{(X_i^* - A_i^*)}{\sigma_{XA}} \cdot \frac{(Y_i^* - B_i^*)}{\sigma_{YB}}. \tag{5.3}$$

(see Klein *et al.*, 2014), with X_i^*, Y_i^*, A_i^* and B_i^* as in (5.2). Contrary to (5.2), in the denominator

$$\sigma_{XA}^2 = 1/(n-1)\sum_{i=1}^{n}(X_i^* - A_i^*)^2 \quad \text{and} \quad \sigma_{YB}^2 = 1/(n-1)\sum_{i=1}^{n}(Y_i^* - B_i^*)^2$$

are calculated across all loci, and therefore, do not depend on i.

Differences between values corresponding to different collectives have been proposed as a base measure for models and algorithms in univariate analyses before (see, e.g., Tarazona *et al.*, 2011, in gene transcription analysis). The scores based on the idea of externally centered correlation proposed here may be seen as a more general version of such measures.

In the remainder of this chapter, the scores are used as a base measure in Bayesian mixture models. In one application, the consistency between ChIP measurements based on microarray vs. sequencing technologies is investigated (Section 5.3). Further, an integrated analysis of gene transcription and histone modifications is carried out on several data sets (Section 5.4). While in Section 5.3, a standard Bayesian mixture model fitting a mixture of normal distributions is discussed in combination with measure (5.2), in Section 5.4 the model employs a mixture of distributions of different types in combination with measure (5.3).

5.3 Validation of ChIP-seq vs. ChIP-chip Techniques

The growing importance of next-generation sequencing in omics research and the growing shift of attention from microarrays to sequencing technology is accompanied by multiple expectations of the scientific community, in particular due to the elimination of restrictions regarding the investigated loci inherent to microarrays. However, as in the context of microarrays, the validity of results obtained with sequencing technology depends as much on laboratory work processes and preprocessing pipelines as on the employed technology itself. These routines usually need some time to be established and respected for a new technology, and their complexity may be higher for more advanced technologies. Given this and seeing the notable amount of research that has been done and will continue to be done with microarray technology for at least some time, a considerable interest lies in across-platform-reliability of results obtained by microarray and sequencing technology. This creates the need for integrative analyses which assess the consistency of results from microarray platforms and next-generation sequencing when targeting the same molecular biologic questions.

While there are software platforms and published work flows capable of

analysing both ChIP-chip and ChIP-seq data, such as Chen *et al.* (2011), Johannes *et al.* (2010), Ji *et al.* (2008) or Zhu (2013), there has been little interest in terms of statistical approaches designed to analyze both inputs jointly in a model-based way. Rather, the two inputs are often integrated with simple means, e.g. in Ho *et al.* (2011), a correlation analysis is carried out to compare results between the two platforms. An exception is the HMM framework of Choi *et al.* (2009), in which ChIP-chip and ChIP-seq are each represented as an individual HMM, whose hidden states in turn are modeled as the bivariate emission probabilities of an additional 'master-level' HMM.

In general, microarray and sequencing techniques render values that are not directly comparable: microarrays produce continuous intensity values for probe sets of determined locations, while sequencing yields positions of particular bound sequences, which only taken together imply a notion of intensity. For a joint analysis, thus, first it has to be counted how many sequences overlap a given location or interval. Still, equal values given by the two methods will not have the same interpretation. A correlation analysis in this situation may help to assess whether there is dependence between the two inputs. However, in this thesis it is argued that the concept of externally centered correlation is more appropriate, jointly measuring the deviations between two collectives in ChIP-chip and ChIP-seq inputs instead of just the dependence between both.

The content of Section 5.3 is the subject of Schäfer *et al.* (2012), which for the most part is based on the Bachelor thesis of Otgonzul Lkhagvasuren (Lkhagvasuren, 2010). The author of this PhD thesis co-supervised the Bachelor thesis and proposed its concept regarding the statistical methodology.

5.3.1 BCR-ABL Data Set

In Section 5.3, the inter-method reliability of ChIP-chip and ChIP-seq technologies is assessed based on a data set consisting of murine hematopoietic 32D cells that were retrovirally stably transduced at the University Hospital of Mün-

ster with the oncogene BCR-ABL, the hallmark of chronic myeloid leukemia (Elling *et al.*, 2011). BCR-ABL is a fusion oncoprotein originating from the genomic translocation t(9;22) and is found in human patients suffering from different types of leukemia, in particular chronic myeloid leukemia but also, e.g., Philadelphia-positive acute lymphoblastic leukemia. Activation or inhibition of genes by histone modifications induced by this oncogene are considered to be essential for understanding these diseases.

To generate reference samples, 32D cells are mock-transduced with the empty vector. The resulting reference cell line is treated with a growth factor in order to assimilate the reference cells to the cancer cells with respect to growth behavior. Chromatin immunoprecipitation with an antibody against acetylated histone 3 at lysines 9 and 14 (anti- H3K9Ac/K14Ac) is carried out, and DNA sequences bound by the antibody are mapped to the genome separately by ChIP-chip and ChIP-seq methods after polymerase chain reaction (PCR).

In the case of ChIP-seq, the DNA sequences are determined by high-throughput screening employing the Illumina Genome Analyzer IIx. In order to map the sequences, the short 35-base-pair sequences are compared to the Mus musculus reference genome. A special interest lies in regions that concentrate many DNA sequences, since it can be assumed that the DNA fragments corresponding to these sequences were bound to histones selected by the antibody during ChIP, and that in consequence the histones were acetylated in this region. In the case of ChIP-chip, DNA fragments are labeled with fluorescent dye and then hybridized to the Affymetrix GeneChip Mouse Promoter 1.0R array. This array contains approximately 4.5 million 25-mer probes (i.e., probes with a length of 25 base pairs) covering more than 25 000 genomic regions near the genes' transcriptional start sites. For each of the probes, an intensity value is measured by a scanner. This measurement process is repeated three times on each cell, using the mean across the repeated measurements as input measure for the analysis. Thus, technically, the resulting database consists of only two cell

lines, including the reference cell line.

5.3.2 Data Preprocessing

On the ChIP-chip data, quantile normalization (Bolstad *et al.*, 2003) is performed using the `R` package `Starr` (Zacher *et al.*, 2010). For the ChIP-seq data, 5 452 964 reads obtained from the BCR-ABL sample and 3 666 697 reads obtained from the reference sample could be aligned to the reference. Duplicate sequences are removed since they likely arise as artifacts during PCR. This reduces the number of reads to 5 111 088 and 3 398 454, respectively, leading to a higher number of reads per chromosome in the BCR-ABL sample compared to the reference sample. This higher number of reads is due only to the higher sequencing depth in the BCR-ABL sample, but not to biological conditions of interest. Thus, to not bias the results, further reads in the BCR-ABL sample are randomly deleted to get an equal number of reads per chromosome in both samples, mimicking equal sequencing depths. The Burrows–Wheeler Alignment tool (Li and Durbin, 2009) is employed to align sequences, using Mus musculus NCBI m37 as reference genome.

The matching of ChIP-chip and ChIP-seq values is carried out at the level of microarray probe sets. Specifically, every ChIP-chip intensity value is assigned the amount of ChIP-seq sequences overlapping its probe set, resulting in one continuous intensity value and one integer number per locus. The length of the original DNA fragments in the experimental setup lies between 100 bp and 1 000 bp with a mean length of more than 200 bp. However, only the first 35 bp at the 5 prime end of the DNA fragments can be sequenced due to technical limitations of the employed sequencing technology. Therefore, the sequences are artificially extended toward the 3 prime end to a total length of 200 bp, resulting in more overlapping sequences. Still, some of the sequences are excluded from the analysis as a result of the matching. Moreover, the sequencing depth in this study is relatively small. Thus, for loci that neither in the BCR-ABL nor in the

reference cell line present bound sequences, it is uncertain whether the absence of such sequences is due to an absence of acetylation or, rather, due to lack of sequencing depth. Such loci are therefore omitted for subsequent analysis.

5.3.3 Approach

For two reasons, it appears reasonable to consider region specific levels of variability within the observed z_i values. The first reason is inherent to ChIP analysis in general. While approximately 150 base pairs of DNA wrap around one histone, the DNA is cut into pieces of a size between 100 and 1 000bp before adding antibodies for acetylated histones in ChIP analysis. For this reason, DNA fragments enriched by the ChIP procedure – i.e. an increased amount of DNA fragments bound to modified histones – can still be found at up to roughly 900 bp distance from an acetylated histone. Although small DNA pieces are better enriched than large ones, to some extent the signals can be assumed to be dependent within this region when assessing externally centered correlation between two ChIP platforms. The second reason is related to the data set. The array probes, used as elementary units in the analysis, are located at distances of approximately only 50 bp within windows around the promoters. Due to this close proximity of probes, signals from adjacent probes are likely to be particularly dependent. These two reasons suggest that measure (5.2), which may take such dependence into account regarding variability, is a good base measure for analysis.

The random variables $Z_1, \ldots, Z_n$ arising from ascertainment of this measure for a number of genomic loci are assumed to be an i.i.d. sample of a random variable Z. The distribution of Z is viewed as consisting of three clusters: a large probability mass is expected to be centered around zero, corresponding to transcripts displaying differences between the two conditions/treatments of interest in none or just one of both data types. A smaller probability mass is expected on the positive axis, corresponding to transcripts displaying equally

directed differences between the two conditions in gene transcription and histone modification data. Finally, an also smaller probability mass is assumed to lie on the negative axis, corresponding to transcripts displaying unequally directed differences between the two conditions in both data types. The difficulty lies in discriminating Z_i values whose deviation from zero is small enough to be explained by random variability from the others.

The argument made here is that a Bayesian mixture model applied to the observed z_i is able to accomplish this task. The model presented subsequently roughly follows the framework presented by Wei and Pan (2008) for their standard normal mixture model and classifies the loci to three groups. Based on such a classification, the reproducibility of results between ChIP-chip and ChIP-seq technologies can be judged. The mixture model for the distribution F of Z with density f is defined as follows:

$$Z_i|\boldsymbol{\mu}, \boldsymbol{\sigma}^2, \boldsymbol{\pi}^* \sim F(\boldsymbol{\mu}, \boldsymbol{\sigma}^2, \boldsymbol{\pi}^*), \tag{5.4}$$

$$f(z_i|\boldsymbol{\mu}, \boldsymbol{\sigma}^2, \boldsymbol{\pi}^*) = \sum_{k=1}^{K} \pi_k^* \cdot f_N(z_i|\mu_k, \sigma_k^2) \quad \text{for} \quad i = 1, \ldots, n, \tag{5.5}$$

with mean vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_K)'$, variance vector $\boldsymbol{\sigma}^2 = (\sigma_1^2, \ldots, \sigma_K^2)'$, and the vector of mixture weights $\boldsymbol{\pi}^* = (\pi_1^*, \ldots, \pi_K^*)'$, where $\sum_{k=1}^{K} \pi_k^* = 1$. $f_N(z_i|\mu_k, \sigma_k^2)$ denotes the density of the normal distribution with mean μ_k and variance σ_k^2, representing the kth group. The parameters π_k^*, σ_k^2 and, partly, μ_k are unknown and estimated by the model. Inverse gamma distributions are assigned to the variances of all normal distributions, σ_k^2, $k = 1, \ldots, K$, while the means, μ_k, $k = 1, \ldots, K$, are either fixed at zero or given a censored normal distribution,

$$\begin{aligned}
\mu_k &\sim N(m_{\mu_k}, v_{\mu_k})I(a, 0) \text{ for } k = 1, \ldots, K/2 - 1, \\
\mu_k &\equiv 0 \text{ for } k = K/2, K/2 + 1, \\
\mu_k &\sim N(m_{\mu_k}, v_{\mu_k})I(0, b) \text{ for } k = K/2 + 2, \ldots, K, \\
\frac{1}{\sigma_k^2} &\sim \text{Gamma}(a_{\sigma_k}, b_{\sigma_k}) \text{ for } k = 1, \ldots, K,
\end{aligned}$$

with $a = \min(z_i)$ and $b = \max(z_i)$ and $N(\mu, \sigma^2)I(c_l, c_r)$ denoting a normal distribution censored to have values between c_l and c_r. The gamma distribution is parameterized as in Section 4.5. Following Wei and Pan (2008), the censored normals are employed to avoid labeling switching between the normal mixing components (Wei and Pan, 2008, speak of truncation in their work, although they implement a censoring). Censoring is employed instead of truncation for computational convenience and its easier implementation in the software WinBUGS, while the impact of the extra probability mass assigned to the value zero under censoring appears to be controlled by the μ_k fixed to zero.

The classification of transcripts is carried out by means of an allocation variable T which is given a categorical distribution. Correspondingly, $T_1, \ldots, T_n$ are the classifications for all transcripts $i = 1, \ldots, n$. The components $k = 1, \ldots, K/2-1$ represent negative Z_i values, the components $k = K/2+2, \ldots, K$ represent positive Z_i values and the components $K/2$ and $K/2+1$ represent Z_i values equal or near to zero. The mixture weights are following a Dirichlet distribution,

$$\begin{aligned} T_i &\sim \text{Categorical}(\pi_1^*, \ldots, \pi_K^*), \\ \pi_1^*, \ldots, \pi_K^* &\sim \text{Dirichlet}(\alpha_0, \ldots, \alpha_0). \end{aligned}$$

For the parameters of the mixing components, prior distributions are chosen inspired by those used in Wei and Pan (2008) and Broet and Richardson (2006). However, more informative priors for the means are chosen to ensure contiguous groups when classifying positive and negative Z_i values. Specifically, it is set $m_{\mu_k,0} = -10$ for $k = 1, \ldots, K/2-1$ and $m_{\mu_k,0} = 10$ for $k = K/2+2, \ldots, K$. For all groups, $k = 1, \ldots, K$, $v_{\mu_k,0} = 1$ and $a_{\sigma_k,0} = b_{\sigma_k,0} = 0.1$ are chosen. For the mixture weights, a uniform Dirichlet prior, $\alpha_0 = 1$, is chosen following Wei and Pan (2008).

5.3.4 Results

Due to memory limitations in fitting the Bayesian model with the software WinBUGS, the analysis is restricted to chromosome 9, which showed a high number of overlapping sequences in preliminary analyses. This results in a total amount of 104 672 loci.

To fit model (5.5) to ChIP-chip and ChIP-seq measurements, $K = 8$ components are chosen. Three of them are earmarked to represent differences between the BCR-ABL cell line and the reference cell line that have the same direction in ChIP-chip and ChIP-seq measurements. Three further components are employed to represent unequally directed differences, while the remaining two components are thought to represent the z_i observations that are either exactly zero or too small in their absolute value to differ significantly from zero. Preliminary analyses showed that the model needs five additional components compared to the three employed by Wei and Pan (2008) to have the necessary flexibility for achieving a classification with desirable characteristics, i.e. three contiguous classes with relatively little overlap.

In particular, the second component for zero values is needed since after removing loci without any mapped sequences in the ChIP-seq data, a considerable proportion of z_i values, 12.16%, is still exactly zero. The additional two components for positive and negative z_i values each are necessary because the distribution of the Z values has both notably longer and thicker tails than the corresponding distribution in Wei and Pan (2008). However, for interpretative purposes the components are grouped into three classes (positive/negative/zero or near zero). In cases with an even greater range of z_i values than in this data set, more additional components may be necessary.

To fit the model, MCMC methods are used employing the software WinBUGS (Spiegelhalter *et al.*, 2003), see Appendix C.4 for the code. For all other types of calculations and analyses, the statistical software environment `R`, version 2.14.1 (R Core Team, 2011), is employed.

Since, as discussed in Section 5.3.3, DNA fragments enriched by the ChIP procedure can still be found at up to roughly 900 bp distance from an acetylated histone, a window size of 1 800 bp is selected in (5.2) for standardization. Non-valid z_i values due to, e.g., the denominator being equal to zero in the ChIP-seq part of (5.2), are removed from the analysis. The posterior distribution of the model parameters is obtained via MCMC methods based on 500 posterior samples, using 75 000 iterations after a burn-in of 2 000 iterations and a thinning of 150. The performance of the MCMC algorithm with respect to convergence and mixing was assessed by examination of trace plots. Burn-in and thinning were specified to ensure that the analyzed sample stems from the desired posterior distribution (trace plots are documented in Figure A.15 in the Appendix). The mean posterior estimates for all model parameters, their standard deviations and the number of assigned loci are given in Table 5.1. In Figure 5.2A and B, the overall fit of the model to the empirical distribution of observed z_i values is plotted. In the histogram, there are visible peaks at –1 and 1 arising due to a number of observations for which the corresponding locus itself is the only one to fall into the standardizing window. This causes the nominator and the denominator to be equal in terms of absolute values. As intended, components 1 to 3 represent the negative z_i values, while components 6 to 8 represent the positive z_i values. Component 5 represents the values that are exactly zero and component 4 captures values that are close to zero.

Every locus is ultimately assigned to one component by its median a posteriori t_i value. The resulting classification is visualized in Figure 5.2C. It can be seen that the model achieves a good separation between the positive, negative and zero z_i values, with only two extreme negative values resulting as 'misclassified' in the sense of producing non-contiguous classes. A high proportion of 87.3 % of all loci is classified to one of the zero components. However, given the unimodal and near-symmetric form of the distribution, this comes not as a surprise (in a case of a more asymmetric or multimodal distribution, more values would be expected to differ notably from zero).

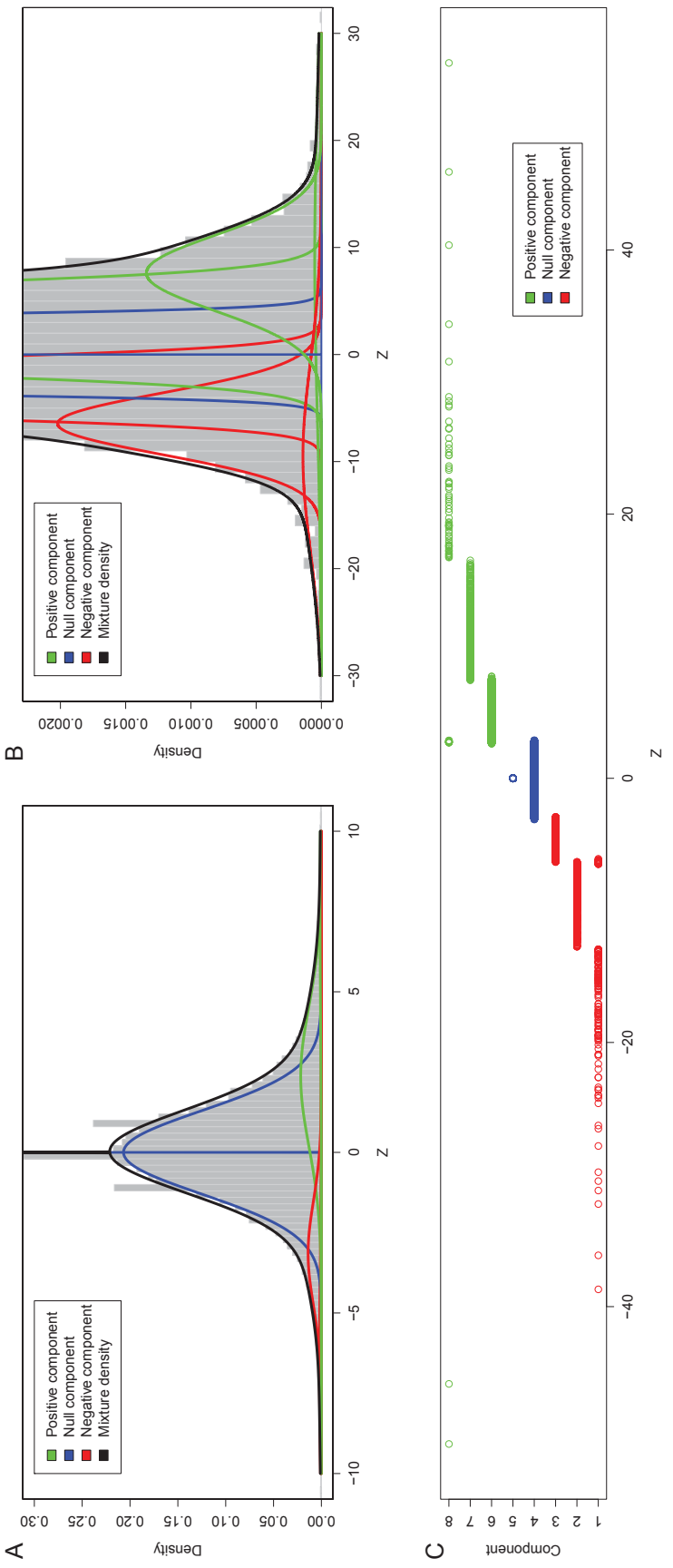

Figure 5.2: Model fit and classification for the BCR-ABL data set as achieved by the Bayesian mixture model employing normal components. The grey histograms in Figure **A** and **B** show the distribution of the observed z_i values at different scales. The histograms are overlaid with the mixture density and the densities of the single components. In Figure **C**, the observed values z_i are plotted against their classification in the mixture model. For better visibility of the model fit, the histograms have been truncated on the y-axis, on the left side due to the huge amount of zero values.

	Negative			Null		Positive		
k	1	2	3	4	5	6	7	8
$\hat{\mu}_k$	-9.53	-6.47	-3.14	0	0	2.37	7.61	8.94
	[-11.46,-7.51]	[-7.86,-4.93]	[-3.43,-2.82]	-	-	[1.79,2.95]	[6.18,9.01]	[6.47,11.14]
$\hat{\sigma}_k^2$	75.03	8.22	2.58	1.68	0.00002	4.71	12.53	217.07
	[24.29,145.0]	[4.08,12.65]	[1.68,3.66]	[1.61,1.76]	[0.00000,0.00002]	[3.56,5.95]	[7.08,18.33]	[96.76,400.2]
$\hat{\pi}_k^*$	0.003	0.01	0.06	0.67	0.12	0.12	0.01	0.002
	[0.000,0.003]	[0.00,0.01]	[0.00,0.06]	[0.63,0.71]	[0.12,0.12]	[0.09,0.15]	[0.01,0.02]	[0.001,0.003]
		0.073			0.795		0.131	
		[0.063,0.086]			[0.759,0.833]		[0.102,0.164]	
Number of genes	185	980	4 367	78 064	13 278	6 949	763	86
		5 532			91 342		7 798	
Proportion	0.002	0.009	0.042	0.746	0.127	0.066	0.007	0.001
		0.053			0.873		0.074	

Table 5.1: Estimated parameters of the Bayesian mixture model for the BCR-ABL data set, employing a mix of normal distributions. The 95% credible intervals are given in square brackets. The curly brackets summarize the weights π_k^*, the number of classified genes or the proportion of classified genes for the three groups, respectively.

One reason for the high number of loci classified to one of the zero components may be that the same antibody (anti-H3K9Ac/K14Ac) was used for both the BCR-ABL and reference cell line, possibly leading to overly similar ChIP-seq profiles in both. The proportions of z_i values classified as positive and negative are of central interest in the analysis: while a proportion of 7.4 % of all loci is classified to one of the positive components, only 5.3 % of loci are classified to one of the negative components. Unfortunately, since the classification is the result of averaging across the posterior, no credible intervals can be calculated for the classification. For comparison, the sum of the mixture weights of the negative components is estimated as $\sum_{k=1}^{3} \hat{\pi}_k^* = 0.073$, whereas the sum of the mixture weights of the positive components is estimated as $\sum_{k=6}^{8} \hat{\pi}_k^* = 0.131$. The latter proportion is nearly twice as big as the first one, and, more importantly, the 95% credible intervals for $\sum_{k=1}^{3} \hat{\pi}_k^*$ and $\sum_{k=6}^{8} \hat{\pi}_k^*$ are [0.063, 0.086] and [0.102, 0.164], respectively, i.e. they do not overlap.

Considered jointly with the classification proportions, this result indicates that ChIP-chip and ChIP-seq technologies agree more than they disagree on the analyzed data. It may be noted that the proportions of loci classified to the positive and negative components are smaller than the sums of the mixture weights corresponding to positive and negative components. The reason is that notable parts of the distributions of components 3 and 6 are masked by the distribution of component 4. The relatively small difference between positive and negative components in classification suggests caution regarding the conclusions of this analysis.

The analysis demonstrates the utility of Bayesian methodology in comparisons between two sources of data in case of small samples sizes. Both model-based estimates and classification results help to interpret the empirical distribution of z_i values. The implications of the results are moderately positive concerning the consistency of important measurement methodologies. However, discrepancies of ChIP-seq and ChIP-chip results occur at an amount of loci that is undesirably high, compared to the amount of loci classified as show-

ing agreement between the two technologies. As a means of validation, the Pearson correlation coefficient between ChIP-seq and ChIP-chip signals is calculated across loci within 1 kb bins, similar to Ho *et al.* (2011). An average correlation of 0.041 is noted on chromosome 9, excluding bins that contain no sequences in one of the cell lines. Thus, the results based on the z_i values reflect to some extent other assessments of association carried out on the ChIP-seq and ChIP-chip profiles.

However, caution is suggested by other works reporting an overlap of about 60% for ChIP-enriched regions determined separately by ChIP-chip and ChIP-Seq (Robertson *et al.*, 2007). A number of factors indicate that generalization of the results presented here concerning the proportion of discordances between ChIP-seq and ChIP-chip technologies should be limited. First, the reference cell line probes were originally treated with a growth factor for ChIP-seq, but without a growth factor for ChIP-chip. Since different experimental conditions affect the comparability between the technologies, several months later new ChIP-chip profiles were generated, this time including the treatment with a growth factor. This may possibly have caused batch effects, whose impact in histone modification analyses remains to be studied. The growth factor, on the other hand, apart from making the cells more similar with respect to their growth behavior, may also have an influence on other cell characteristics. Second, since the sequencing depth in this study is relatively small, a number of low- and medium-size peaks in the sequencing data may go undetected. Finally, Ho *et al.* (2011) suggest that the cross-platform validity of ChIP-chip and ChIP-seq measurements depends heavily on the used antibody. In particular for anti-H3K9Ac, an antibody similar to the one used in the experiments of this study, they obtain the worst results in a comparison of several antibodies.

In conclusion, it cannot be ruled out that the experimental conditions harm the results in this analysis. One likely source of bias, the choice of an algorithm to perform peak calling on the ChIP-seq data (cf. Ho *et al.*, 2011), is, however, eliminated in this analysis since peak calling is not carried out.

5.4 Integration of Histone Modification and Gene Transcription Data

The main focus in this section lies on integrative analysis of histone modifications and gene transcription data. This is a classic research question in epigenetics since epigenetic modifications associated with a change in gene transcription are more likely to belong to driver gene, i.e. to have a causal influence on an investigated condition such as the development of cancer. Particularly, transcripts showing differences between normal and cancerous samples that are equally directed in both an activating histone modification and gene transcription are suspected of being driver transcripts. Transcripts showing differences between conditions that are unequally directed in both a repressive histone modification and gene transcription are attributed a potential role in cancer as well.

The content of Section 5.4 is the subject of Klein *et al.* (2014). In this work, the author of this PhD thesis and Hans-Ulrich Klein are joint first authors. While the author of this PhD thesis developed, implemented, analysed and discussed the Bayesian mixture model, Hans-Ulrich Klein was responsible for the choice of the data sets, data preprocessing, additional implementations, the simulation study and a part of the biological interpretation.

5.4.1 Data Sets

Four data sets are considered in Section 5.4 and described in the following, three experimental ones and a simulated one. The first experimental data set is obtained from *Cebpa*$^{\text{fl/fl}}$ (hereafter termed *Cebpa* wild-type) and *Cebpa*$^{\text{fl/fl}}$;*Mx1Cre* conditionally deleted for *Cebpa* (hereafter termed *Cebpa* knockout) hematopoietic stem and progenitor cells (LSK cells) (Hasemann *et al.*, 2014). Specimens from three *Cebpa* knockout mice and three wild-type mice are hybridized separately on six Affymetrix Mouse Gene 1.0 ST arrays (Gene Expression Omnibus GSE49975). ChIP against histone H3 lysine K4 trimethylation (H3K4me3) is

applied to two pools of three wild-type mice each and two pools of three knock-out mice each. Eventually, specimens are sequenced on an Illumina Genome Analyzer IIx sequencer producing 36bp reads (GSE43007). The reads are mapped against the reference genome (mus musculus, version mm10) employing the Burrows-Wheeler Aligner (Li and Durbin, 2009).

A second experimental data set is taken from a study investigating epigenetic differences between a prostate cancer cell line (LNCaP) and normal primary prostate cells (PrEC) (Bert *et al.*, 2013). RNA-seq measuring RNA transcription in LNCaP and PrEC cells is carried out by means of an Illumina Genome Analyzer IIx. Using the STAR algorithm (Dobin *et al.*, 2013), the resulting 76bp, single-end reads are aligned against the human reference genome (hg19) annotated with splice junctions obtained from the GENCODE project (Harrow *et al.*, 2012) to improve alignment accuracy. Regarding histone modifications, the activating histone H3 lysine K4 trimethylation (H3K4me3) and the repressive histone H3 lysine K27 trimethylation (H3K27me3) in the LNCaP and PReC cells are mapped by ChIP-seq (Illumina HiSeq 2000 and Illumina Genome Analyzer IIx, respectively, 50bp reads), aligning the reads against the human reference genome (hg19) by means of the Burrows-Wheeler Aligner (Li and Durbin, 2009). The data set can be downloaded at Gene Expression Omnibus (GSE38685).

A further, third experimental data set is obtained from a study (Schenk *et al.*, 2012) comparing the effect of treatment with all-trans-retinoic acid (ATRA) to the effect of a combined treatment of ATRA and trans-2-phenylcyclopropylamine (TCP) on a human leukemic HL60 cell line. Transcription data of two replicates are measured on an Illumina HumanHT-12 v4 Bead Chip and normalized using the method proposed by Lin *et al.*, 2008. ChIP-seq is employed to measure Histone H3 lysine K4 dimethylation (H3K4me2) on an Illumina Genome Analyzer IIx (58bp reads) without replicates. Reads are aligned against the human reference genome (hg19) and can be downloaded together with the normalized transcription data (Gene Expression Omnibus GSE34726).

The simulated data set is generated employing the *Cebpa* data set, the only experimental data set for which replicates are available both in ChIP-chip and ChIP-seq. It may be assumed that between biological replicates, differences exist neither in gene transcription nor in histone modification, thus the replicates are ideal for simulation purposes. The first two knockout mice replicates are taken as base data without differential genes. Differential genes are then generated by multiplying the transcription value and the ChIP-seq value (calculated as described in Section A.9) of the first knockout mice replicate by $(1 + c)$ with $c \in \{-1, -0.9 \ldots, -0.1, 0.1, \ldots, 1\}$. 100 genes with a ChIP-seq value above the median of all ChIP-seq values are randomly chosen for each level of factor c, leading to 2 000 out of 21 236 genes with equally directed differences in both data types.

5.4.2 Data Preprocessing

Contrary to the approach in Section 5.3, the analysis now focuses on the transcript level. The intention of this focus is to summarize the information conveniently for interpretation and to reduce the amount of elementary units, which facilitates the analysis of the whole genome instead of only a small subset like in Section 5.3. Thus, the matching of RNA and ChIP data is performed at the level of transcripts. For this purpose, genomic regions centered at the transcripts' TSSs are defined since the histone modifications studied here primarily occur at TSSs (as only exception, H3K27me3 occurs additionally throughout gene bodies; see Dong *et al.*, 2012). For these genomic regions, a measure for the abundance of a transcript or group of transcripts from either RNA-seq data or gene expression microarray data is obtained. Further, a ChIP-seq value based on the number of ChIP-seq reads aligned within the genomic region of that transcript is calculated.

The software STAR used to align RNA-seq reads is capable of producing spliced alignments. Resulting abundances are reported in fragments per kilobase

of transcript per million fragments mapped (FPKM). To obtain FPKM values of a certain magnitude, they are multiplied with the ratio of the sample's 0.75 quartile to the average 0.75 quartile value across all samples, as implemented in the Cuffdiff software (Trapnell *et al.*, 2013). FPKM values of transcripts sharing the same TSS are summarized to obtain a single value for each TSS. Finally, FPKM values are logarithmized. In the following, in case of RNA-seq data, X_i and A_i denote the normalized FPKM values of the two considered conditions (e.g., cancer and normal cells, respectively) for TSS $i = 1, \ldots, n$.

In contrast to RNA-seq or ChIP-seq, gene expression microarrays render continuous measurement values that can be directly assigned to transcripts based on the given array design. Several normalization methods for various array platforms exist. For Affymetrix gene expression arrays, the established method robust multi-array Average (RMA, see Irizarry *et al.*, 2003) is applied to the raw mRNA intensities prior to analysis. In RMA, first a background correction is conducted that filters out unspecific hybridization as well as noise. Subsequently, quantile normalization (Bolstad *et al.*, 2003) is performed to remove further biases. For Illumina Bead Chips, a variance-stabilizing transformation (Lin *et al.*, 2008) is carried out. In case of all methods, a logarithmic or similar transformation is applied on the transcription values.

In case of array data, X_i and A_i denote the normalized transcription values of the two considered conditions/treatments measured at probe $i = 1, \ldots, n$.

ChIP-seq values are obtained for a given transcript i by counting the number of sequenced fragments that lie within the genomic region $\mathcal{R}_i$ of width w centered at its TSS. The length of the sequence reads equals roughly the length necessary to locate the reads in the genome and is much shorter than the length of the real DNA fragments. To assess overlaps in a more realistic way, thus, reads are expanded towards the 3 prime end to the mean DNA fragment length (here 200bp to 350bp). Let Y_i and B_i denote the resulting number of reads that overlap $\mathcal{R}_i$ in the two considered conditions. The choice of a reasonable size w for $\mathcal{R}_i$ depends on the studied histone modification. For many histone

modifications, an appropriate size is known (Hebenstreit *et al.*, 2011) and in the following analyses the choice of w is not crucial for the chosen approach. Alternatively, a proper choice of $\mathcal{R}_i$ could be obtained from applying a peak detection algorithm on the ChIP-seq data. In case of RNA-seq data, X_i and A_i are matched directly to Y_i and B_i. In array technology, however, there is not necessarily a one-to-one relation between array probes and transcripts. Several probes may correspond to one transcript, and, more importantly for matching in this case, probes may measure several transcripts with different TSSs. In such cases, $\mathcal{R}_i$ is defined as the union of the regions derived for single transcripts measured by probe i. Thus, $\mathcal{R}_i$ may consist of more than one genomic interval and its size may differ for different probes.

To account for different total number of reads and different ChIP efficiency, quantile normalization (Bolstad *et al.*, 2003) is applied to the ChIP-seq values. When maximizing the correlation between the differences in the transcription data and the differences in the ChIP-seq data, this method performs better than linear scaling based on the total number of reads as used by many peak detection algorithms, e.g. MACS (Zhang *et al.*, 2008), or than scaling based on the median of count ratios as proposed by Anders and Huber (2010) for RNA-seq data (see Appendix A.9 as well as Figures A.16, A.19 and A.24 for more details on normalization methods and results, respectively). Alignment statistics about the numbers of mapped reads for the RNA-seq and ChIP-seq data sets can be found in Tables A.12, A.13 and A.14, Appendix A.10.

The ChIP-seq values are ordered, so that $\mathsf{Y}_{\iota_1} \leq \ldots \leq \mathsf{Y}_{\iota_n}$ and $\mathsf{B}_{\tau_1} \leq \ldots \leq \mathsf{B}_{\tau_n}$. The normalized values are then defined as $Y_{\iota_i} = B_{\tau_i} := (\mathsf{Y}_{\iota_i} + \mathsf{B}_{\tau_i})/2$.

5.4.3 Approach

In this section, as opposed to Section 5.3, only one of the two inputs represents histone modification measurements. Furthermore, the analysis is carried out at the level of transcripts, leading to considerably greater physical distance be-

tween the observations than in Section 5.3, where research was done at the level of probes. For these two reasons, now there is less potential for region specific levels of variability in the observed z_i values. Consequently, a genom-wide normalization is chosen instead of the normalization within windows around the TSS, leading to (5.3) as appropriate measure. Thus, the variances of the differences, $\sigma^2_{XA} = 1/(n-1)\sum_{i=1}^{n}(X_i - A_i)^2$ and $\sigma^2_{YB} = 1/(n-1)\sum_{i=1}^{n}(Y_i - B_i)^2$, are calculated across all transcripts. If there are several biological replicates in the data set, an average is eventually used in (5.3) after matching and normalization, i.e. if m ChIP-seq replicates are available, $Y_i = 1/m\sum_{j=1}^{m} Y_{ij}$, where Y_{ij} is the quantile normalized ChIP-seq value of the jth replicate for transcript i.

As in Section 5.3, if a transcript presents equally directed differences in transcription and histone modification data, its observed z_i value is positive, while unequally directed differences lead to a negative z_i value. Thus, for an activating histone modification, the distribution of Z is expected to be slanted towards positive values. If for a transcript differences exist only in one of the considered variables or there are no differences at all, the Z_i value is expected to be close to zero. This case is assumed to be the most frequent one, so the distribution of Z is expected a priori to feature a large probability mass centered around zero. Transcripts displaying unequally directed differences between the two conditions in both variables are likely represented by a smaller probability mass on the negative axis. The transcripts displaying equally directed differences between the two conditions in gene transcription and histone modification data, of most interest in the analysis, will likely correspond to an also smaller probability mass on the positive axis.

In building a model for the Z_i, the goal is to discriminate between these three classes of values. The two extreme classes of transcripts with either high or low Z_i values are expected to be easily separated with an appropriate model. The challenge, thus, lies in correctly identifying Z_i values differing little enough from zero to be explained by random variability. An intuitive approach to analyze the distribution of Z reflecting this conception would be a three-component mixture

model, i.e. a model representing each of the three transcript classes by one component. However, a class may be represented by more than one component in a mixture model when convenient for achieving a good fit and classification. In Section 5.3, a mixture model with eight Gaussian components represents three clusters because less components, while preferable for interpretation, would not provide enough flexibility in terms of fit. In this section, a more general modeling approach is proposed, potentially reducing the number of necessary mixture components. This leads to advantages when, e.g., considerable probability mass in the tails of the Z distribution constitute a challenge to standard models.

In Taslim *et al.* (2009), a mixture model is fitted considering not only normal distributions, as in traditional model-based clustering, but both normal and exponential distributions. While a number of normal components represent the center of the distribution, the probability mass in the distributions' tails is covered by two exponential distributions, one of which is mirrored at zero. Each exponential component potentially replaces several normal components, i.e. the number of components can be considerably reduced compared to a standard normal mixture model. To determine the number of normal components necessary to represent the center of the distribution in such a model, possible criteria are the fit and, more importantly, whether a clean classification is achieved in the sense of producing three contiguous domains of values.

In the following, the idea of Taslim *et al.* (2009) to mix normal and exponential distributions will be applied in a Bayesian context. Formally, Z is assumed to be a random variable and $Z_1, \ldots, Z_n$ to be an i.i.d. random sample of Z. The mixture model for the distribution F of Z with density f is defined as follows:

$$\begin{aligned} Z_i|\boldsymbol{\lambda}, \boldsymbol{\sigma}^2, \boldsymbol{\pi}^* &\sim F(\boldsymbol{\lambda}, \boldsymbol{\sigma}^2, \boldsymbol{\pi}^*), \\ f(z_i|\boldsymbol{\lambda}, \boldsymbol{\sigma}^2, \boldsymbol{\pi}^*) &= \pi_1^* \cdot f_{exp}(-z_i|\lambda_1) \\ &\quad + \sum_{k=2}^{K-1} \pi_k^* \cdot f_N(z_i|\sigma_k^2) \\ &\quad + \pi_K^* \cdot f_{exp}(z_i|\lambda_K), \quad i = 1, \ldots, n, \end{aligned} \tag{5.6}$$

with the vector of exponential parameters $\boldsymbol{\lambda} = (\lambda_1, \lambda_K)'$, variance vector $\boldsymbol{\sigma}^2 = (\sigma_2^2, \ldots, \sigma_{K-1}^2)'$ and vector of mixture weights $\boldsymbol{\pi^*} = (\pi_1^*, \ldots, \pi_K^*)'$, where $\sum_{k=1}^K \pi_k^* = 1$. $f_N(z_i|\sigma_k^2)$ denotes the density of the normal distribution with mean 0 and variance σ_k^2, representing the kth group, and $f_{exp}(z_i|\lambda_k)$ denotes the density of the exponential distribution with parameter λ_k. Gamma hyperprior distributions are assigned to the precisions of the normal distributions, $1/\sigma_k^2$, $k = 2, \ldots, K-1$, and to the parameters λ_1 and λ_K of the two exponential distributions:

$$\begin{aligned} \frac{1}{\sigma_k^2} &\sim \text{Gamma}(a_{\sigma_k}, b_{\sigma_k}) \text{ for } k = 2, \ldots, K-1, \\ \lambda_k &\sim \text{Gamma}(a_{\lambda_k}, b_{\lambda_k}) \text{ for } k = 1, K, \end{aligned}$$

where the gamma distribution is parameterized as in Section 4.5. These are the respective conjugate prior distributions, allowing to employ an efficient Gibbs sampler. However, the density function of the exponential distribution is strictly monotone and has only one parameter to simultaneously specify both its mean and variance. Thus, an increase of probability mass at large Z_i values automatically leads to a shift of probability mass near zero as well. This is an undesirable feature, since the classification of observations with large Z_i values should not put structural constraints on the classification of observations with a small Z_i value. For this reason, gamma distributions are considered as an alternative to exponential distributions, leading to the alternative model

$$\begin{aligned} Z_i|\tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\beta}}, \boldsymbol{\sigma^2}, \boldsymbol{\pi^*} &\sim \widetilde{F}(\tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\beta}}, \boldsymbol{\sigma^2}, \boldsymbol{\pi^*}), \\ \widetilde{f}(z_i|\tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\beta}}, \boldsymbol{\sigma^2}, \boldsymbol{\pi^*}) &= \pi_1^* \cdot f_{gam}(-z_i|\tilde{\alpha}_1, \tilde{\beta}_1) \\ &\quad + \sum_{k=2}^{K-1} \pi_k^* \cdot f_N(z_i|\sigma_k^2) \\ &\quad + \pi_K^* \cdot f_{gam}(z_i|\tilde{\alpha}_K, \tilde{\beta}_K), \quad i = 1, \ldots, n, \end{aligned} \tag{5.7}$$

in the same notation as (5.6), where the vectors $\tilde{\boldsymbol{\alpha}} = (\tilde{\alpha}_1, \tilde{\alpha}_K)'$ and $\tilde{\boldsymbol{\beta}} = (\tilde{\beta}_1, \tilde{\beta}_K)'$ contain the shape and scale parameters of the two gamma distributions, re-

spectively, and $f_{gam}(z_i|\tilde{\alpha}_k, \tilde{\beta}_k)$ denotes the density of the gamma distribution with parameters $\tilde{\alpha}_k$ and $\tilde{\beta}_k$. Again, gamma distributions are assigned to the precisions of the normal distributions, $1/\sigma_k^2$, $k = 2, \ldots, K-1$. The gamma hyperparameters $\tilde{\alpha}_k$ and $\tilde{\beta}_k$, $k = 1, K$ are assigned gamma distributions as well:

$$\begin{aligned}
\frac{1}{\sigma_k^2} &\sim \text{Gamma}(a_{\sigma_k}, b_{\sigma_k}) \quad \text{for } k = 2, \ldots, K-1, \\
\tilde{\alpha}_k &\sim \text{Gamma}(a_{\tilde{\alpha}_k}, b_{\tilde{\alpha}_k}) \quad \text{for } k = 1, K, \\
\frac{1}{\tilde{\beta}_k} &\sim \text{Gamma}(c_{\tilde{\beta}_k}, d_{\tilde{\beta}_k}) \quad \text{for } k = 1, K.
\end{aligned}$$

For $\tilde{\alpha}_k$, there is no conjugate prior distribution. Thus, a Metropolis-Hastings step is implemented for sampling, employing a normal proposal distribution centered at the actual value and mirrored at 0, with a standard deviation adjusted to achieve a convenient acceptance rate.

Regardless of whether exponential or gamma distributions are employed, the allocation variable T is given a categorical distribution, and $T_1, \ldots, T_n$ are the classifications for all transcripts $i = 1, \ldots, n$. The mixture weights are assigned a finite dimensional Dirichlet prior, as advocated by Ishwaran and Zarepour (2002a) and as employed in integrative genomics by, e.g., Kirk *et al.* (2012),

$$\begin{aligned}
T_i &\sim \text{Categorical}(\pi_1^*, \ldots, \pi_K^*), \\
\pi_1^*, \ldots, \pi_K^* &\sim \text{Dirichlet}(\alpha^*/K, \ldots, \alpha^*/K).
\end{aligned}$$

The assignment of transcripts to the components depends essentially on the concentration parameter α^*. Thus, it is assigned a prior distribution as well and estimated within the model. As common in the literature (see, e.g. Ishwaran and Zarepour, 2002a), a gamma prior is chosen,

$$\alpha^* \sim \text{Gamma}(a_{\alpha^*}, b_{\alpha^*}).$$

The model is fitted with MCMC methods using a Gibbs sampler (see Ishwaran and James, 2001, and Neal, 2000, for details of the implementation see also Sections A.8.1 and A.8.2 in the Appendix) and model estimates are obtained by

calculating the mean across the posterior distributions. For α^*, exact sampling within the Gibbs sampler is not possible as it would be, e.g., in the truncated finite approximation to the Dirichlet process (Ishwaran and Zarepour, 2000). Consequently, a Metropolis-Hastings step is again employed here in analogy to the shape parameter of the gamma distributions, and again the proposal distribution is adjusted to achieve a convenient acceptance rate.

Concerning the joint analysis of the gene transcription and ChIP-seq histone modification data sets, in preliminary analyses it was found that the minimum number of normal components to achieve both a good overall fit and three contiguous classes is four, leading to a total number of six components. The normal components representing the observed Z values equal to or near zero are given a fixed mean $\mu_k = 0$ and a small variance a priori. The parameters of the exponential components are assigned a small value a priori, resulting in a large mean. This ensures that these components represent the tails of the Z distribution, avoiding label switching between the components corresponding to distinct classes. Analogously, the parameters of the gamma distributions are chosen to result in a large mean a priori as well. Specifically, while $a_{\sigma_k,0} = b_{\sigma_k,0} = 10$ for $k = 2, \ldots, K-1$, for $k = 1, K$ the prior values $a_{\lambda_k,0} = b_{\lambda_k,0} = 0.005$, $a_{\tilde{\alpha}_k,0} = b_{\tilde{\alpha}_k,0} = 2$ and $c_{\tilde{\alpha}_k,0} = d_{\tilde{\alpha}_k,0} = 0.1$ are chosen. The standard deviation of the Metropolis-Hastings step for the gamma shape parameter $\tilde{\alpha}_k$ is chosen as 0.025. For the Dirichlet parameter, $a_{\alpha_0^*} = 2$ and $b_{\alpha_0^*} = 0.5$ are chosen as proposed by Ishwaran and Zarepour (2000) and the standard deviation of the Metropolis-Hastings step is chosen as 1.

The approach is implemented in the `R` package `epigenomix` (Klein and Schäfer, 2014) available via the BioConductor project (URL: http://www.bioconductor.org), see also Section C.5 in the Appendix for documentation.

5.4.4 Results on *CEBPA* Knockout Data Set

There are $i = 1, \ldots, 21\,236$ transcription values x_i^* and a_i^* for both conditions, obtained by averaging across the wild-type and knockout replicate samples, respectively, for each probe annotated with at least one transcript. 7 163 probes are assigned to several transcripts with different TSSs according to the ENSEMBL data base (Flicek *et al.*, 2013). Based on work by Hebenstreit *et al.* (2011), as well as on visual inspection of the read distribution at TSSs, a width of $\varpi = 6\,000$ is chosen for the promoter region $\mathcal{R}_i$ when calculating ChIP-seq values y_i^* and b_i^*, $i = 1, \ldots, 21\,236$. In order to assess the effect of different choices of ϖ, the Pearson correlation ρ between $x_i^* - a_i^*$ and $y_i^* - b_i^*$, $i = 1, \ldots, 21\,236$, is calculated for different choices of ϖ ranging from 100 bp to 20 000 bp (Figure A.16A, Appendix A.10). The choice of ϖ seems not to be crucial, unless it is chosen too small. Furthermore, the correlation for probes with a unique TSS ($\rho = 0.200$) does not differ remarkably from probes with multiple TSSs ($\rho = 0.203$), indicating that the aggregation of reads from multiple TSSs is suitable. Figure A.16A also shows that quantile normalization of the ChIP-seq data leads to a higher correlation between the differences in the transcription data and the differences in the ChIP-seq data than several other studied normalization methods.

Figures 5.3A and B show the fit of the model given in (5.6) to the observed z_i values. Figure 5.3C shows the classifications obtained from 100 000 iterations using every 10th iteration after a burn-in of 50 000 iterations (this configuration is also used for the other data sets). Parameter estimations and classifications are given in Table 5.2 and trace plots in Figure A.17, Appendix A.10. The acceptance rate in the Metropolis-Hastings step is 0.395. Acceptance rates are documented in the results tables for all data sets. Classifications are obtained by calculating the mode of t_i values across MCMC iterations. The mixture distribution fitted to the z_i has more probability mass at the right tail than at the left tail, as reflected by the estimated weights $\hat{\pi}_k^*$ of the mixture components:

while the weight of the negative component $\hat{\pi}_1^* = 0.001$ is negligible, the positive component's weight is estimated as $\hat{\pi}_6^* = 0.037$. Only 17 transcripts are classified to the negative component. 20 809 are classified to the null components, while 410 transcripts are classified to the positive components and are thus potential drivers. Thus, as expected, the majority of transcripts does not show differences in both data types. However, the transcripts with differences in both data types reveal that an increase (decrease) in H3K4me3 is associated with an increase (decrease) in gene transcription.

Among the 410 transcripts in the positive class, there are several genes that have been implicated in hematopoietic stem cell biology (*Mecom*, *Kit*) and/or acute myeloid leukemia (AML) (*Pim1*, *Pim2*, *Hoxa9*, *Meis1*), highlighting the functions of *Cebpa* in normal and malignant hematopoiesis. All of these mentioned genes have a z_i value greater than 5. The transcript coding for the hepatic leukemia factor *Hlf* has by far the highest z_i value among all transcripts ($z_i = 110.8$) and among humans is commonly associated with Acute lymphoblastic leukemia (ALL) rather than with AML. Specifically, there are chromosomal translocations fusing the *E2A* gene with either *Hlf* or the *Pbx1* gene that result in the generation of chimeric proteins which in turn may cause ALL (Aspland *et al.*, 2001). It has been shown that the translocation protein *E2A-Pbx1* resulting of the fusion of *E2A* and *Pbx1* causes AML in mice (Kamps and Baltimore, 1993), so a new hypothesis arising from the analysis may be that this is also the case for the *E2A-Hlf* fusion protein.

KEGG pathway analyses (Dennis *et al.*, 2003) identify the pathways *Acute myeloid leukemia* and *JAK-STAT signaling pathway* as significantly enriched (FDR < 0.05). The involvement of the pathway *Acute myeloid leukemia* is plausible since mutations of the *Cebpa* gene have been considered as a prognostic factor in acute myeloid leukemia (AML). Specifically, double mutations in *Cebpa* have been linked to a good outcome (Fasan *et al.*, 2014). The *JAK-STAT signaling pathway* has been attributed a role in the pathogenesis of AML as well: the Janus kinases (JAK) and further subsequent elements of the pathway,

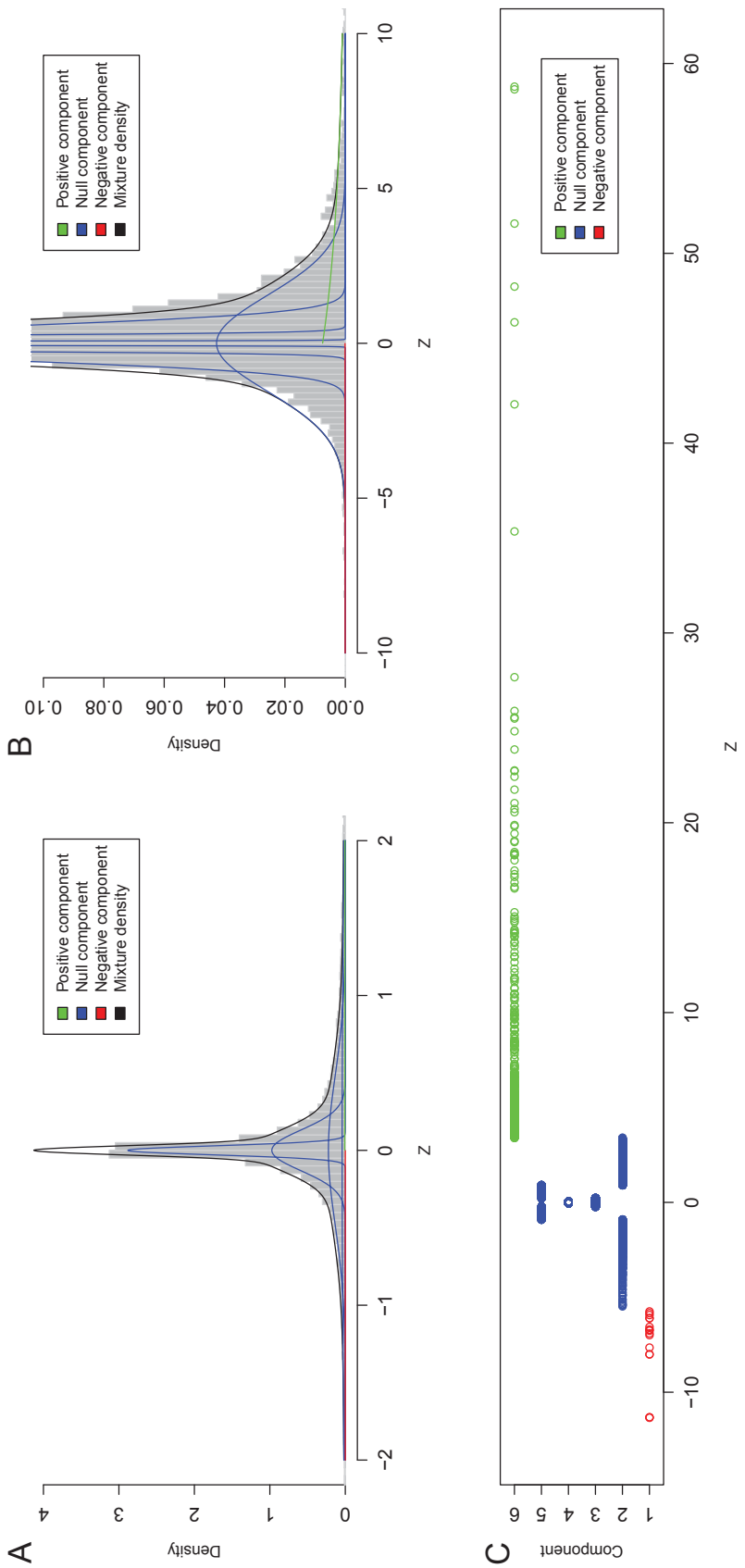

Figure 5.3: Model fit and classification for the *Cebpa* knockout data set, employing a mix of normal and exponential distributions. The grey histograms in Figure **A** and **B** show the distribution of the observed z_i values at different scales. The histograms are overlaid with the mixture density and the densities of the single components. In Figure **C**, the z_i are plotted against their classification in the mixture model. One transcript classified to component 6 with $z_i \approx 110$ is omitted. More transcripts are classified into the positive than into the negative component indicating a positive correlation between H3K4me3 modifications and gene transciption.

	Negative	Null				Positive
k	1	2	3	4	5	6
$\hat{\sigma}_k^2$	-	2.395	0.017	0.001	0.227	-
	-	[2.008,2.858]	[0.015,0.020]	[0.001,0.001]	[0.178,0.282]	-
$\hat{\lambda}_k$	0.072	-	-	-	-	0.203
	[0.031,0.120]	-	-	-	-	[0.179,0.227]
$\hat{\pi}_k^*$	0.001	0.165	0.319	0.210	0.267	0.037
	[0,0.002]	[0.144,0.186]	[0.297,0.340]	[0.195,0.226]	[0.248,0.285]	[0.031,0.043]
		0.962 [0.955, 0.968]				
Number of genes	17	2 541	7 221	6 054	4 993	410
		20809				
Proportion	0.001	0.120	0.340	0.285	0.235	0.019
		0.980				

Table 5.2: Estimated parameters of the Bayesian mixture model for the *Cebpa* knockout data set, employing a mix of normal and exponential distributions. The 95% credible intervals are given in square brackets. The curly brackets summarize the weights π_k^*, or, respectively, the number of classified transcripts for all four null components and the corresponding proportions. The acceptance rate for the Metropolis-Hastings steps for α^* in the MCMC run is 0.3954.

e.g., signal transducer and activator of transcription (STAT) factors, are involved in aspects of cancer development such as cellular proliferation and differentiation (Lee *et al.*, 2013). These findings indicate that the results of the analysis are biologically meaningful.

To investigate whether the results are reproducible, two data sets are generated containing one microarray and one ChIP-seq replicate from the knockout and wild-type collective each. Both data sets thus consist of a single biological entity. The model fits and transcript classifications that result from a separate analysis of each set of replicates are shown in Figure 5.4. As can be seen in the contingency table of the classification results (Table 5.3), in each of the two separate analyses less transcripts are classified to the positive component than in an analysis of the entire data set. Possibly, this is due to the variances σ^2_{XA} and σ^2_{YB} being smaller in the entire data set as a result of the averaging across the replicates. There are 94 transcripts that are classified to the positive component in both separate analyses, but only one transcript is classified to the negative component in both cases.

To assess the specificity of the approach, an analysis is conducted in which the two compared conditions do not present biological differences. Specifically, the first and second *Cebpa* wildtype replicates are compared. Due to the absence of biological differences between these samples, all transcripts are expected to be classified to the null components. Indeed, the weights $\hat{\pi}^*_1 = 0.000001$ and $\hat{\pi}^*_6 = 0.0004$ (Figure 5.4C) of the two exponential components are very small. Only 5 out of 21 236 transcripts were assigned to the positive component, and no transcript was assigned to the negative component (Figure 5.4F).

Finally, the potential advantages of employing gamma distributions instead of exponential ones in the mixture model are explored. Figures 5.5A and B show the fit of the model (5.7) to the z_i values, and Figure 5.5C the obtained classifications. Parameter estimations and classifications are given in Table 5.4. Trace plots are given in Figure A.18, Appendix A.10.

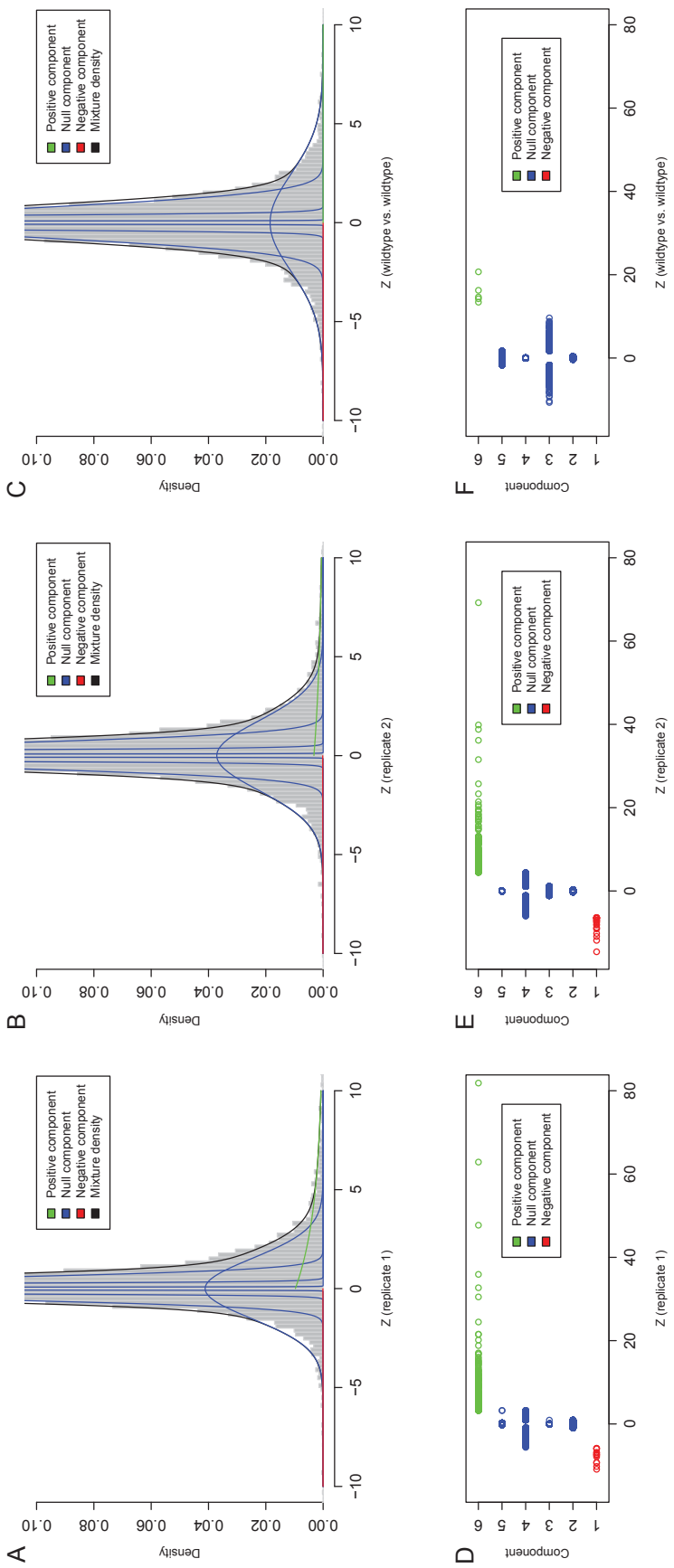

Figure 5.4: Model fits and classifications for subsets of the *Cebpa* knockout data set, employing a mix of normal and exponential distributions. Figure **A** (**B**) shows the distribution of the observed z_i values and the mixture model's fit when comparing the first (second) *Cebpa* knockout replicate to the first (second) wild-type replicate. The corresponding classifications are given in Figures **D** and **E**, respectively. Figures **C** and **F** present the mixture model comparing the first to the second *Cebpa* wildtype replicate.

		Replicate 2			
		Negative	Null	Positive	Sum
Replicate 1	Negative	1	11	1	13
	Null	20	20 722	115	20 857
	Positive	2	270	94	366
	Sum	23	21 003	210	

Table 5.3: Classification results of both replicates from the *Cebpa* knockout data set, employing a mix of normal and exponential distributions. Shown are the number of transcripts classified into the negative, null and positive components when applying the Bayesian mixture model separately to the first and second set of biological replicates.

Again, as reflected by the estimated weights $\hat{\pi}_k^*$ of the mixture components, the mixture distribution fitted to the z_i has more probability mass at the right tail than at the left tail. However, compared to the model employing exponential distributions, this model tends to allocate more transcripts to the negative and positive components: the corresponding weights $\hat{\pi}_1^* = 0.033$ and $\hat{\pi}_6^* = 0.111$ are notably bigger. Now, 103 transcripts are classified to the negative component, 20 151 to the null components and 850 to the positive component.

While these differences are not big enough to alter the overall general interpretation, some observations are worthwhile. As expected, the greater flexibility of the gamma distribution translates into a better model fit. The exponential distribution with parameter λ^* is equivalent to a gamma distribution with shape parameter 1 and scale parameter $1/\lambda^*$. Since the estimates for the shape parameters of the gamma distributions are approximately 0.4, an added value is visible here in terms of flexibility. Visually, a slightly better fit of the mixture distribution to the histogram of z_i values can also be observed when comparing Figure 5.3B and Figure 5.5B, in particular in the left tail of the distribution. However, there is a downside of this increased flexibility. First, the classification does no longer result in strictly contiguous classes, as a number of values close to 0 are allocated to the components that correspond to the gamma distributions.

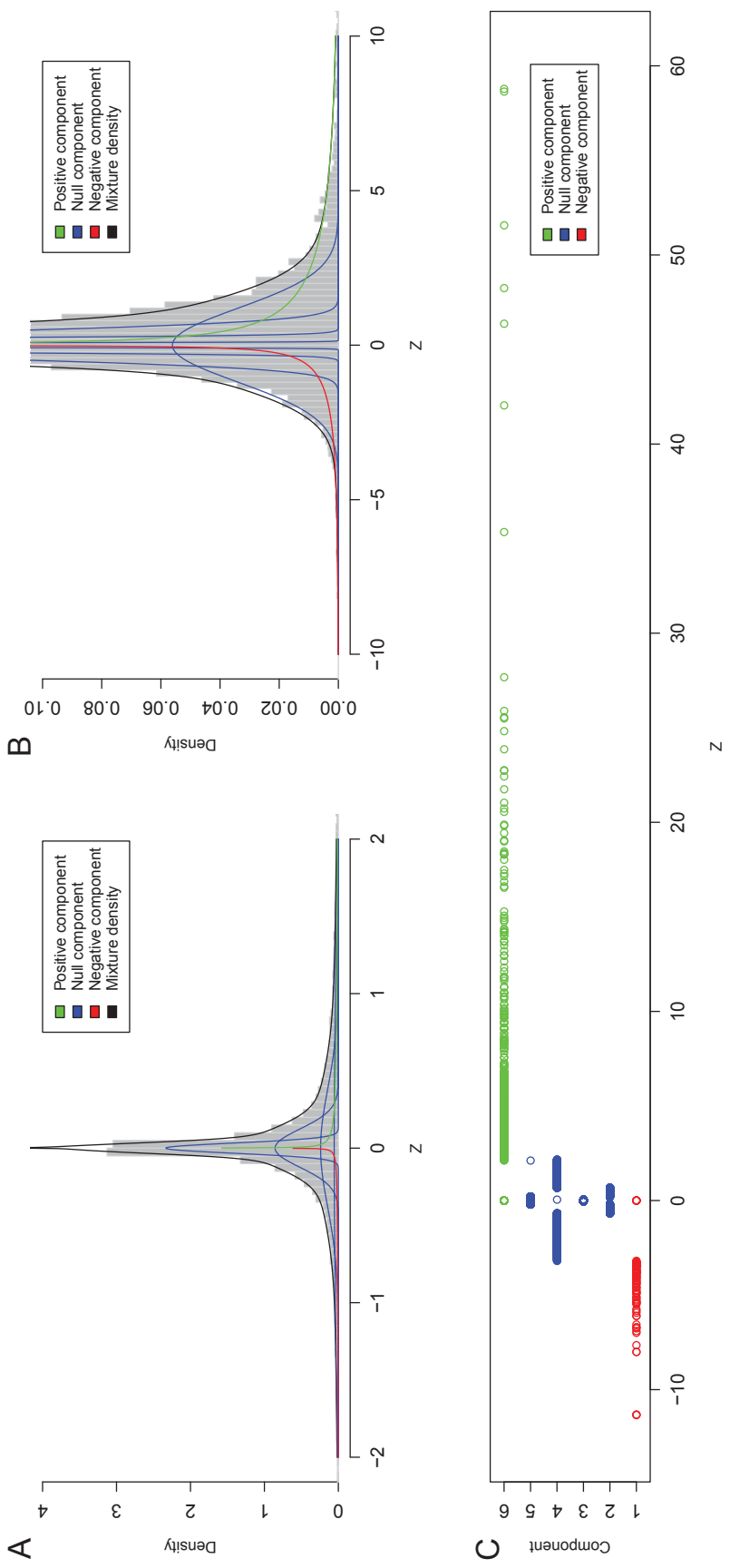

Figure 5.5: Model fit and classification for the *Cebpa* knockout data set, employing a mix of normal and gamma distributions. The grey histograms in Figure **A** and **B** show the distribution of the observed z_i values at different scales. The histograms are overlaid with the mixture density and the densities of the single components. In Figure **C**, the observed values z_i are plotted against their classification in the mixture model. One transcript classified to component 6 with $z_i \approx 110$ is omitted. More transcripts are classified into the positive than into the negative component, indicating a positive correlation between H3K4me3 modifications and gene transcription.

	Negative	Null				Positive
k	1	2	3	4	5	6
$\hat{\sigma}_k^2$	-	0.144	0.001	1.43	0.016	-
	-	[0.104,0.194]	[0.001,0.001]	[1.142,1.749]	[0.012,0.019]	-
$\hat{\tilde{\alpha}}_k$	0.407	-	-	-	-	0.411
	[0.343,0.481]	-	-	-	-	[0.383,0.441]
$\hat{\tilde{\beta}}_k$	2.76	-	-	-	-	5.061
	[2.145,3.684]	-	-	-	-	[4.476,5.716]
$\hat{\pi}_k^*$	0.033	0.228	0.193	0.168	0.266	0.111
	[0.020,0.050]	[0.202,0.253]	[0.176,0.211]	[0.146,0.190]	[0.233,0.296]	[0.097,0.128]
		0.855 [0.824, 0.881]				
Number of genes	102	4 516	6 220	3 037	6 379	850
		20152				
Proportion	0.005	0.214	0.295	0.144	0.302	0.040
		0.955				

Table 5.4: Estimated parameters of the Bayesian mixture model for the *Cebpa* knockout data set, employing a mix of normal and gamma distributions. The 95% credible intervals are given in square brackets. The curly brackets summarize the weights π_k^*, or, respectively, the number of classified transcripts for all four null components and the corresponding proportions. The acceptance rates for the Metropolis-Hastings steps for α^*, $a_{\tilde{\alpha}_k}$ and $c_{\tilde{\beta}_k}$ in the MCMC run are 0.577, 0.466 and 0.291, respectively.

Second, the reasonable specificity achieved by the model employing exponential distributions is largely lost. When comparing conditions that do not present biological differences, still a notable amount of transcripts is classified to the positive and negative components. This can be concluded from results comparing the first and second *Cebpa* wildtype replicates as given in Figure A.27 in Appendix A.10. The loss of specificity is a considerable disadvantage and greatly reduces the models's utility in practice. For this reason, in the remainder of the thesis, the focus is put on the model based on a mixture of normal and exponential components. Results for all data sets and models not shown in the main document are nevertheless documented in Appendix A.10.

5.4.5 Results on Prostate Cancer Data Set

Of a total of 113 663 transcripts or groups of transcripts sharing the same TSS, 46 657 show transcript abundances in both LNCaP and PrEC cells and are used for analysis (based on annotation from the GENCODE project, Harrow *et al.*, 2012). For both considered histone modifications, H3K4me3 and H3K27me3, a smaller promoter width than for the *Cebpa* knockout data set is chosen with $\varpi = 3\,000$. This decision is based on the correlation observed for different choices of ϖ (see Figure A.19 in Appendix A.10) between differences in ChIP-seq values and transcription values. Further, it takes into account the fact that unlike gene expression microarrays, RNA-seq allows to distinguish between different transcripts of the same gene whose TSSs are in close proximity. The correlation of $\rho = 0.289$ observed for H3K4me3 is higher than in the *Cebpa* knockout data set, indicating that either a larger fraction of transcripts shows epigenetic and transcriptional differences when comparing prostate cancer cells to normal cells, or these differences are larger now. For the repressive histone modification H3K27me3, a negative correlation is observed as expected ($\rho = -0.197$).

The application of the Bayesian mixture model to the z_i values calculated from data on gene transcription and activating histone modification (H3K4me3)

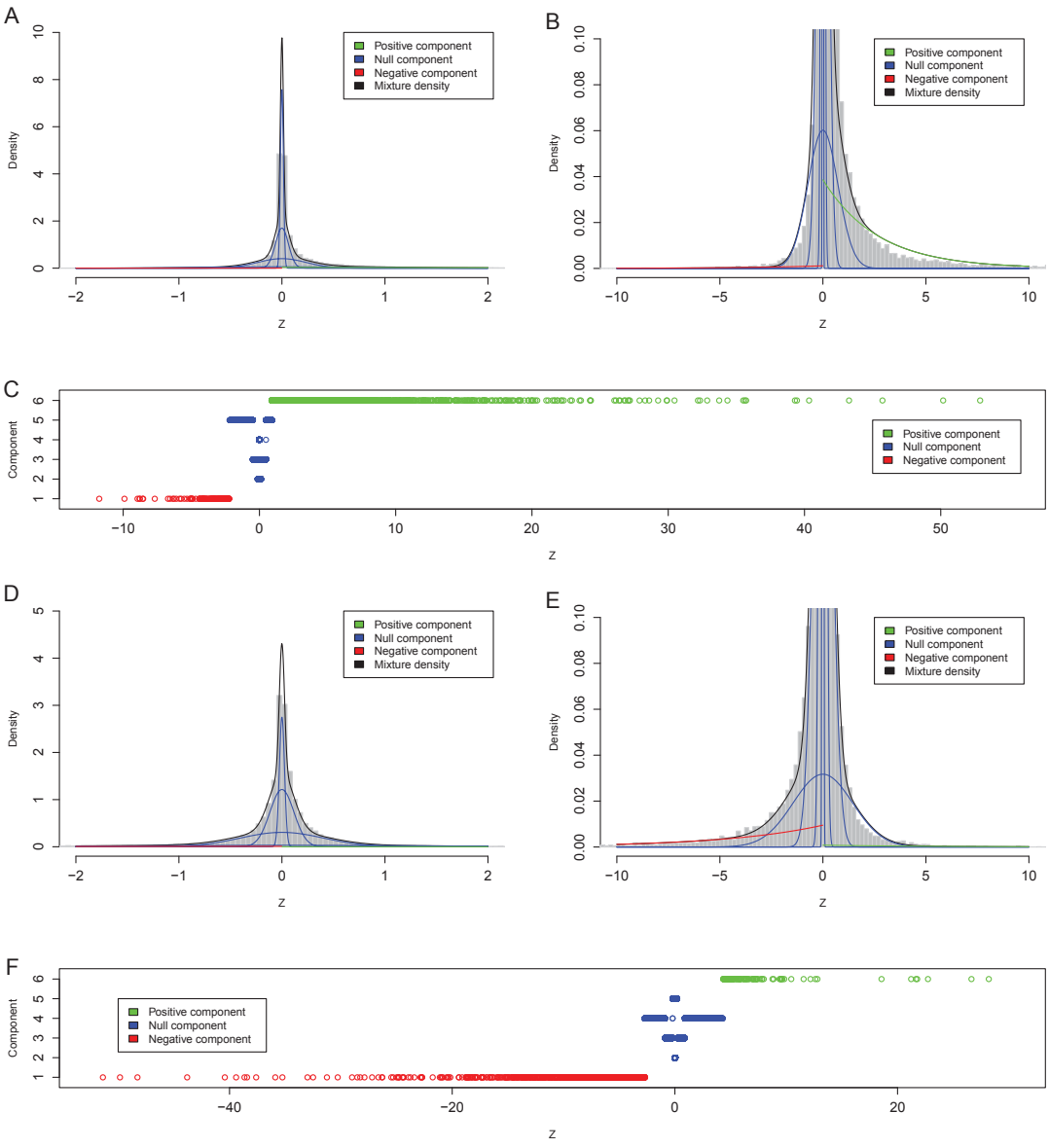

Figure 5.6: Model fit and classification for the prostate cancer data set, employing a mix of normal and exponential distributions. The two grey histograms **A** and **B** show the distribution of the observed z_i values at different scales after matching the transcription data to the H3K4me3 ChIP-seq data. The histograms are overlaid with the mixture density and the densities of the single components. In Figure **C**, the observed z_i are plotted against their classification. Figures **D**, **E** and **F** are the corresponding plots when using the ChIP-seq data for the H3K27me3 histone modification.

results in 3 526 transcripts assigned to the positive and 272 transcripts assigned to the negative component (see Table A.15 in Appendix A.10). Correspondingly, a small weight of $\hat{\pi}_1^* = 0.007$ for the negative component and a notably bigger weight of $\hat{\pi}_6^* = 0.107$ for the positive component are observed (Figure 5.6A-C and Table A.15, Appendix A.10). Trace plots are given in Figure A.20, Appendix A.10. The KEGG (Dennis *et al.*, 2003) pathways *Focal adhesion*, *Axon guidance* and *Pathways in cancer* are found to be significantly enriched (FDR < 0.05) among the transcripts classified as positive, i.e. as potential drivers. According to recently published work, axon guidance genes play a role in pancreatic carcinogenesis (Biankin *et al.*, 2012) and possibly are involved in other cancers as well (Mehlen *et al.*, 2011). Alternative promoter usage is observed for only 8 genes involving 20 transcripts, and, importantly, transcripts of genes classified to component 6 are mostly up or down regulated in a consistent way. For instance, *TPM1* and *SMTN* have multiple promoters and all of their transcripts are classified to component 6. Some of their transcripts display negative differences between the data sets in both gene transcription and histone modification, while some transcripts present positive differences in both inputs. Both genes are involved in actin cytoskeleton development and stabilization.

For the repressive histone modification H3K27me3, weights of $\hat{\pi}_1^* = 0.060$ ($\hat{\pi}_6^* = 0.015$) are estimated and 2 098 (436) transcripts are classified to the negative (positive) component (see Figure 5.6D-F and Table A.16, Figure A.20 in Appendix A.10). When carrying out a pathway analysis on the transcripts now classified as negative, the pathways *Focal adhesion*, *Axon guidance* and *Pathways in cancer* are again significant like in the H3K4me3 analysis (FDR < 0.05). The fact that 732 transcripts are both assigned to the negative component in the H3K27me3 analysis and classified to the positive component in the H3K4me3 analysis reflects interactions between occurrences of active H3K4me3 and repressive H3K27me3 marks.

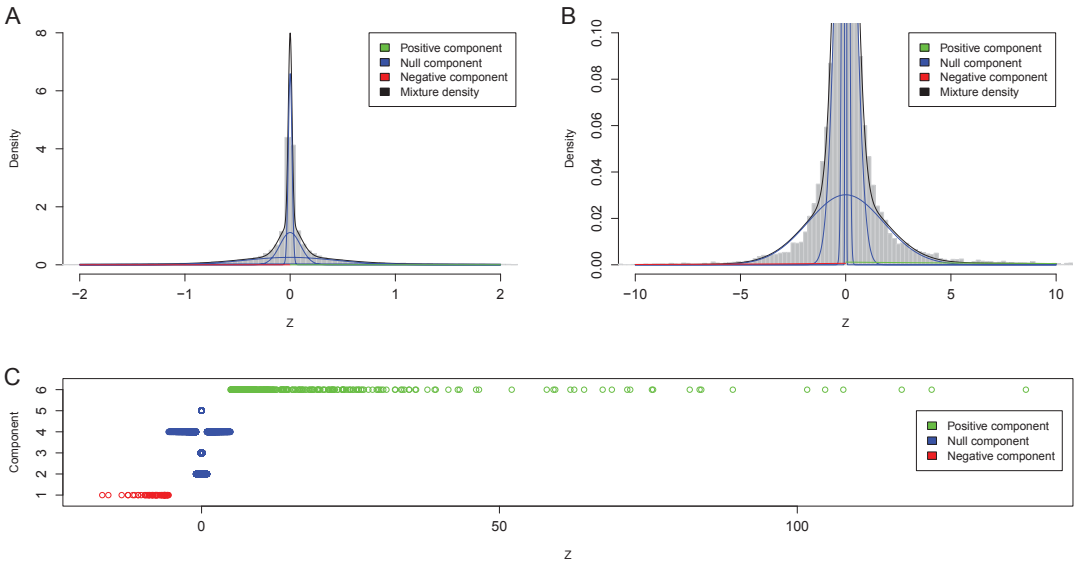

Figure 5.7: Model fit and classification for the ATRA and TCP treatment data set, employing a mix of normal and exponential distributions. The grey histograms **A** and **B** show the distribution of the observed z_i values at different scales and are overlaid with the mixture density and the densities of the single components. In Figure **C**, the z_i are plotted against their classification.

5.4.6 Results on ATRA/TCP Data Set

Of 31 753 probes that can be annotated with at least one transcript, 19 813 are assigned to several transcripts with different TSSs. The promoter width is chosen as $w = 10\,000$ for this data set (Hebenstreit *et al.*, 2011), an assessment of the correlation of the differences between the two treatments in gene transcription and ChIP-seq values is shown in Figure A.24 in the Appendix.

When applying the Bayesian mixture model to the z_i values calculated for the ATRA/TCP data set, a weight of $\hat{\pi}_1^* = 0.004$ for the mirrored exponential component and a weight of $\hat{\pi}_6^* = 0.015$ for the positive component are observed (Figure 5.7 and Table A.17, Figure A.26, in Appendix A.10). Correspondingly, 64 genes are classified to the negative and 349 genes to the positive component. Although the weight of the positive component is higher than the one of the negative component, the quotient of the two weights is not as big as for the other data sets (except for the one with the repressive histone modification

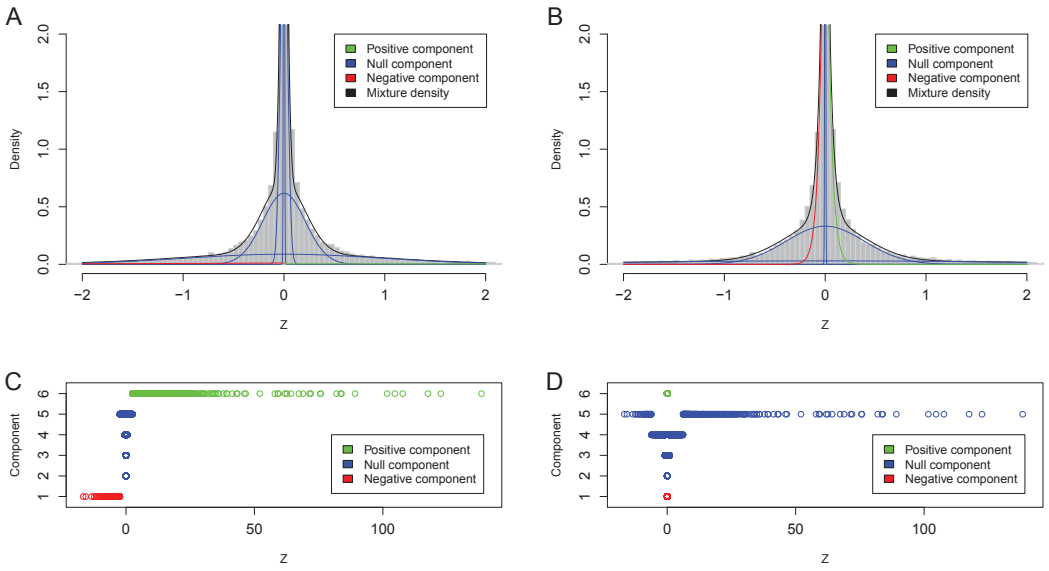

Figure 5.8: Model fit based on maximum likelihood estimations for the ATRA and TCP treatment data set, employing a mix of normal and exponential distributions. (**A**) The expectation maximization (EM) algorithm was applied with $\sigma_2 = 0.1$ $\sigma_3 = 0.5$, $\sigma_4 = 1$, $\sigma_5 = 2$, $\pi^* = (\frac{1}{6}, \ldots, \frac{1}{6})$, $\lambda_{neg} = \lambda_{pos} = 0.1$ as initial values. (**B**) The starting values of λ_{neg} and λ_{pos} were changed to 5. In both cases, the EM algorithm converged in (local) maxima.

H3K27me3). For this reason, it can be concluded that an increase in H3K4me2 may lead to an increase or, in contrast to H3K4me3, also to a decrease in gene transcription. This is in line with previous works showing that H3K4me2 may be present at the promoters of active as well as inactive transcripts (Zhang *et al.*, 2009). Interestingly, many of the transcripts that are classified to the positive component, i.e. identified as potential drivers, belong to genes known to be expressed during hematopoietic stem cell differentiation, such as *IL1B*, *Jak2* or *CD48* (Kiel *et al.*, 2005). These results indicate that the combined ATRA and TCP treatment strengthens cell differentiation effects when comparing it to a treatment based only on ATRA.

The mixture model is further fitted to the ATRA and TCP data set by means of the EM algorithm (like in Taslim *et al.*, 2009) to compare the Bayesian MCMC approach to the standard frequentist fitting technique. As it turns out, the EM algorithm for this data set converges into different local maxima when started

with different initial values (see Figure 5.8). Specifically, the model fit shown in Figure 5.8B does not result in a reasonable classification although it does lead to a higher likelihood than the model fit shown in Figure 5.8A. The allocation of z_i values to the components differs notably between the two sets of initial values: in Figure 5.8A, the values near zero are represented by components 2-5 as intended, while components 1 and 6 represent the negative and positive z_i values, respectively. In Figure 5.8B, however, both positive and negative values are jointly represented by component 5 and the values near zero by the other components. While set (A) of initial values leads to the classification of 1 185 z_i values (3.73%) to components 1 and 6, when employing set (B) of initial values only 328 z_i values (1.03%) are classified to component 5. Thus, by employing set (B) instead of set (A), not only are the components permuted and the negative and positive classifications are no longer distinguishable, but also are the overall proportions notably different.

5.4.7 Results on Simulated Data Set

Analyzing the simulated data set, it is possible to assess for how many transcripts the model delivers a correct classification to either the positive component or one of the other components. Of the 2 000 out of 21 236 transcripts in the simulated data set that display equally directed differences in both data types, the Bayesian mixture model is able to classify 1 656 to the positive component based on the z_i values. A total of 15 transcripts is falsely classified to the positive component and 344 transcripts are falsely classified to a null or negative component, which corresponds to a sensitivity of 0.828 and a specificity of 0.999. To compare the Bayesian mixture model to a naive separate analysis of both data types, a threshold Υ is chosen for the observed differences $x_i^* - a_i$ of the transcription values. All transcripts with $|x_i^* - a_i| \geq \Upsilon$ are considered as differentially transcribed and an equal number of transcripts with the largest absolute differences $|y_i^* - b_i^*|$ of the ChIP-seq values is considered as differentially

factor c	Bayesian mixture model		Naive approach	
	Num. detected	Sensitivity	Num. detected	Sensitivity
$c \in \{-1, 1\}$	200	1	194	0.970
$c \in \{-0.9, 0.9\}$	200	1	186	0.930
$c \in \{-0.8, 0.8\}$	200	1	178	0.890
$c \in \{-0.7, 0.7\}$	196	0.980	181	0.905
$c \in \{-0.6, 0.6\}$	195	0.975	163	0.815
$c \in \{-0.5, 0.5\}$	192	0.960	155	0.775
$c \in \{-0.4, 0.4\}$	182	0.910	118	0.590
$c \in \{-0.3, 0.3\}$	162	0.810	99	0.495
$c \in \{-0.2, 0.2\}$	101	0.505	62	0.310
$c \in \{-0.1, 0.1\}$	28	0.140	29	0.145

Table 5.5: Results from the simulation data set splitted by the magnitude of simulated differences. Data are simulated as described in Section 5.4.1. The fraction c indicates the strength of the simulated differences, e.g., $c = 0.5$ ($c = -0.5$) means that the gene transcription value and the corresponding ChIP-seq value are increased (decreased) by 50% in one of the two replicate samples. For comparability, the threshold of the naive approach is chosen such that it achieves the same specificity (0.999, 15 false positive out of 19 236 true negative genes) as the Bayesian mixture model. Overall, the Bayesian mixture model achieves a sensitivity of 0.828, whereas the sensitivity of the naive separate analyses is 0.683. Notably, the gain in sensitivity of the integrative approach is most distinct for moderate differences ($0.2 \leq c \leq 0.5$). The acceptance rate for the Metropolis-Hastings steps for α^* in the MCMC run is 0.3092.

histone modified. Transcripts both differentially transcribed and differentially histone modified for which $\text{sign}(x_i^* - a_i) = \text{sign}(y_i^* - b_i^*)$ are classified as potential drivers by the naive analysis.

Subsequently, varying thresholds $\mathfrak{t}_{sim}$ are considered and ROC curves are plotted based on these results as well as on the classifications implied by the Z score (see Figure A.30 in the Appendix). At the same level of specificity, the naive approach achieves a sensitivity of 0.683, which is smaller than the value achieved by the Bayesian mixture model. Table 5.5 shows that especially for moderate differences, the Bayesian mixture model achieves a gain in sensitivity of about 0.3 on average.

Chapter 6

Summary and Discussion

In this PhD thesis, finite Bayesian mixture models with a small fixed number of components have been developed to answer actual research questions in two different contexts concerning molecular biophysics (Chapter 4) and molecular biology (Chapter 5).

In Chapter 4, the novel Bayesian approach GAMMICS was introduced. GAMMICS builds on frequentist modeling ideas proposed by Byers and Raftery (1998) for distinguishing large objects from noise and employs them for a Bayesian analysis of small clusters in presence of singletons. Relying on a two-component gamma mixture model for the squared distances of points to their second nearest neighbors, it classifies proteins as either clustered or non-clustered. It is designed to estimate parameters of spatial nanoclustering exhibited by membrane-bound Ras proteins, particularly the proportion of clustered points, the mean cluster size and the mean cluster radius. GAMMICS combines ideas of both single-linkage hierarchical clustering and density-based clustering in a Bayesian framework. Specifically, it estimates the crucial user-defined parameter (the maximum nearest neighbor distance in density-based clustering, the cutoff in the dendrogram in hierarchical clustering) in a Bayesian mixture model and obtains posterior distributions for the cluster size and cluster radius

in an iteration-wise post-processing step. Specifically, in GAMMICS this crucial parameter is a cutoff corresponding approximately to the point of intersection between the two gamma distributions fit in the mixture model.

To compare the approach to other state-of-the-art methods regarding its performance, a comprehensive simulation study was conducted. A point process model was designed for this purpose, the double Matérn cluster process. Unlike standard cluster processes such as the Neyman-Scott process, it allows for non-clustered points and metaclusters, i.e. clusters containing clusters and singletons. Contrary to other recently proposed point process models that have these features (see, e.g., Wiegand *et al.*, 2009), the double Matérn cluster process also allows for clusters outside of metaclusters and permits to freely choose both the proportion of clustered points and the cluster size at each clustering level.

The performance of GAMMICS is generally favorable compared to the common cluster approach DBSCAN, Bayesian model-based clustering employing a Dirichlet process mixture (DPM) model, and a Bayesian version of DBSCAN based on the DPM model. It also outperforms the H-function approach commonly used in biophysical literature to estimate the cluster radius. While the H-function and the two methods based on the nonparametric DPM model nowhere show an outstanding performance, DBSCAN achieves the best median estimation accuracy when estimating the proportion of clustered points. GAMMICS achieves both the smallest median misclassification rate and the smallest median estimation error when estimating the mean cluster radius. In addition, when using one of the two considered procedures to estimate a cluster partition in the post-processing step, GAMMICS also performs best in estimating the mean cluster size.

The DPM model, although it has frequently been used for clustering, is originally a model for density estimation. This may explain its poor performance. Possible reasons for the poor accuracy of the H-function estimates include a greater susceptibility to factors such as drift of the cells during the measure-

ment process, noise inherent in the measurements, or metaclustering. Also, the peak in the H-function is not always sharp. Thus, in some cases when the H-function performs particularly weak, H-function values assumed for a number of radii moderately smaller than the one maximizing the function may be almost as high as the maximum.

While the completely algorithmic DBSCAN performs best for one of the four parameters of interest, it does so only with crucial knowledge about the true clustering parameters. GAMMICS, on the other hand, performs best for the other three without depending on such knowledge and functions with much weaker assumptions regarding distances between points. It also offers a model that quantifies the uncertainty for its estimates, providing more insights and a better interpretability. Contrary to many established cluster algorithms, noise is explicitly considered and quantified.

Taking into account that prior knowledge of Ras clusters is still not taken for granted, the results confirm the advantages of GAMMICS. However, this comes with a cost in terms of computation time: while DBSCAN never takes more than a few seconds to run, GAMMICS typically runs several, sometimes a large number of hours on the data sets in this study. The MCMC routines from the `R` package `DPpackage` and Bayesian DBSCAN are implemented in C, they are thus faster than GAMMICS. However, they may still take several hours depending on the data set. Bayesian DBSCAN consists of several C and `R` routines that have to be applied subsequently and in a manual way to each data set. This makes its application difficult whenever dealing with a large number of data sets.

The relatively weak performance of GAMMICS compared to DBSCAN when estimating the proportion of clustered points may be due to an asymmetric overlap occurring at times between the two posterior gamma distributions. In such cases, the tails of the two gamma distributions that are literally 'cut off' by the calculated GAMMICS cutoff correspond to probability masses of notably different sizes. The estimation of the proportion of clustered points may suffer

a bias in such cases, even if the probability masses corresponding to clustered points and singletons are in principle well separated by the cutoff. Based on typical shapes of the empirical distribution of second nearest neighbor distances, an underestimation of the proportion is more likely than an overestimation.

This potential bias might be attenuated by allocating the proteins to the two components based on whether their second nearest neighbor distance is smaller or larger than the cutoff between the two densities, instead of on the sampling carried out to fit the model. Such an approach, described in Schäfer and Ickstadt (2012), would however lead to an even more algorithmic fashion of the estimation. By consequence, the incorporation of prior knowledge would no longer be possible. Also, in cases when clustered proteins and singletons are not particularly well separated in the histogram of the second nearest neighbor distances, the 'fuzzy separation' of the two groups implied by the mixture model's allocations may be of advantage compared to a strict separation. To perform the algorithmic estimation for the mean cluster size and the mean cluster radius in a more fuzzy fashion as well, one might replace the hierarchical clustering employed for the estimation with a fuzzy version of such clustering (see, e.g., Torra, 2005). The subsequent calculation of mean cluster and mean radius could then take into account the provided cluster membership probabilities, potentially improving estimation accuracies.

In theory, it may happen that the two weighted gamma densities have more than one point of intersection or none at all (without considering 0 and ∞). Also, the two densities might 'switch sides', i.e. the distribution assigned priors for representing the singletons could eventually represent the clustered points, and vice versa. While these cases are improbable in practice, the algorithm calculating the cutoff for the estimation of a cluster partition in GAMMICS is designed to neutralize or at least attenuate a potential bias.

In GAMMICS' mixture model, the two gamma distributions fit to the empirical distributions of distances are weighted by the proportion of clustered points p_c and $1 - p_c$, respectively. In the algorithm that calculates the cluster

partition, these weights are taken into account. Alternatively, it is considered to use the unweighted gamma distributions in this algorithm. This is equivalent to setting $p_c = 0.5$ in the weighted version and, while less intuitive, leads to smaller errors when estimating the mean cluster size and radius. Probably, it makes the procedure less susceptible against artificially extreme estimates for p_c in the mixture model. It is therefore the preferred option.

The GAMMICS method implies the assumption of independence for the squared distances of points to their second nearest neighbors. While appearing justifiable in the interest of model simplicity, this assumption is problematic because the knowledge of the whole point pattern is required to calculate the nearest neighbor distance for any of the proteins. In general, independence of second nearest neighbor distances may be assumed for some point processes depending on the context, e.g., Byth and Ripley (1980) postulate this independence for tree patterns if a random subset of no more than 10% of all trees is considered. However, such an argument depends on the physical characteristics of the considered objects. For trees, a minimum distance between objects can be assumed due to physical restrictions such as branch length and sunlight distribution. While in principle there is a minimum distance between Ras proteins, it is so small in relation to the measurement accuracy that in practice a similar argument cannot be made here. Thus, analyzing subsets of, e.g., only 10% of protein does not necessarily help while discarding most of the information.

Since the cluster radii and sizes are calculated in an algorithmic post-processing step, the inference is not fully model-based. However, due to identifiability issues concerning cluster size and cluster radius, this appears to be the price to pay in exchange for estimating all three parameters without prior knowledge on them. If prior knowledge does exist, it can be used to specify a prior distribution in the usual Bayesian way, but only for the proportion of clustered proteins and not for the cluster radius and the cluster size.

In this thesis, two ways to estimate the radius of a cluster are considered: one half of the maximum distance between two points within the same cluster,

and the mean pairwise distance between points within the cluster, multiplied with the factor $2 \cdot \frac{45\pi}{128}$ of the maximum distance between two points within the same cluster (the factor is based on the theoretical distribution of distances between points spread uniformly in a circle). For each method, the estimator resulting in a better estimation accuracy is chosen. However, other approaches to estimate the cluster radius might be used as well and lead to slightly different results.

GAMMICS relies essentially on dividing the empirical distribution of distances between points into two sets, corresponding to clustered and non-clustered points. Thus, it has weaknesses in detecting the extreme cases in which either no points or all points are clustered. In these cases, the model will still tend to estimate a cutoff although there is none in the data. A reasonable strategy to rule out these cases can make use of Monte Carlo tests based on, e.g., the K-function.

Contrary to most other methods for cluster analysis, the estimation of a posterior cluster partition is not a primary goal in this thesis and, thus, in GAMMICS. However, a posterior cluster partition could be obtained via the posterior similarity matrix based on the partitions inferred in each MCMC step. As in the DPM model, a criterion such as Binder's loss could be utilized for this purpose.

In order to eliminate the algorithmic parts in GAMMICS, one could combine its gamma mixture model with a DPM model within the same MCMC sampling framework. One of the DPM model's weaknesses is that it tends to group too many singletons into clusters, so GAMMICS's gamma mixture model could furnish the DPM model with the classification into singletons and clustered points. Carried out on the points classified as clustered by the gamma mixture model in each iteration, the DPM model may render a more accurate cluster partition and, potentially, estimates for mean cluster size and mean cluster radius equally precise as (or even more precise than) those obtained by the full GAMMICS method. Ji *et al.* (2009), e.g., present a similar combination of a

classification based on two densities representing two classes (although not in form of a formal mixture model) and a DPM model fitted on a set of points updated based on the classification. If, in addition, another DPM model is run on the points estimated as singletons by the gamma mixture model, the resulting two intensities (one for a cluster process and one for a singleton process) might both be compared across experimental conditions and thus help to do inference on the differences of the clustering behavior, e.g., between healthy cells and tumor cells. However, while such approaches would avoid algorithmic elements, they still do not allow to model cluster radius and size directly. To apply the method to experimental data on a large scale, a systematic and automated selection of the regions of interest would be needed, taking into account varying point densities and edge effects.

In this PhD thesis, fixed cells have been investigated in which the Ras proteins are stationary. In future research it will further be of interest to conduct research on living cells with moving proteins and/or on cells with a Ras expression on an endogenous level (corresponding to cells in a normal, healthy state, as opposed to cancerous cells where Ras is overexpressed). The consideration of living cells over time turns the so far spatial problem into a three dimensional one, with time as third dimension. If GAMMICS is combined with an appropriate tracking analysis, providing protein labels over time, it could take into account time as third dimension when calculating second nearest neighbor distances. In this way, clusters would be defined as groups of proteins close in space and time. This would require the absence of blinking among the proteins, since blinking would hamper the tracking analysis. The distance measure used in the model would have to ensure an adequate influence of both space and time domains, e.g., by standardizing or adequately scaling the variables prior to the analysis, or by employing alternatives to Euclidean distance.

Ras interacts with other signaling proteins that possibly modify its cluster characteristics during the course of signal transduction. Alternatively, Ras may influence these binding partners in their role in the further signal transduction

(Tian *et al.*, 2007). Examples of such signaling proteins interacting with Ras include galectines (Hancock, 2006), and Raf kinases, which pass on the signals of active Ras proteins (Wellbrock *et al.*, 2004). It is of interest how the interaction of Ras clusters with such binding partners modifies the subsequent steps in the signaling chain and how this influences the relevant biological processes like cellular growth or tumor progression. To investigate these questions, it is necessary to analyze the joint distribution of Ras and its interacting partners on the nanoscale. In addition, it may be of interest to analyze the joint distribution of different types of Ras proteins that differ in their chemical composition. For these purposes, the experimental setup might be extended to a dual color imaging (Bates *et al.*, 2007; Shroff *et al.*, 2007). Since the output of such a setup would technically allow to discriminate the differently colored proteins from the start, a simple option to assess a joint distribution might be, e.g., to apply GAMMICS to both protein types separately in a unified sampling framework, assuming a priori independent clustering behaviors. Currently, the applicability of GAMMICS and further methods to a joint analysis of two proteins continues to be investigated in the group of Katja Ickstadt based on simulations (Hermann *et al.*, 2015).

In Chapter 5, scores inspired by the externally centered correlation coefficient (Schäfer *et al.*, 2009) were designed to measure congruence between two different omics variables. Congruence in this context is understood as the degree to which transcripts present differences between a target sample and a reference sample that are equally directed in both variables. Only one sample per input and condition is required, facilitating analyses for sequencing data featuring extremely small sample sizes. The score values z_i allow to rank genomic loci such as probes or transcripts regarding their probability of belonging to a driver gene, i.e. of contributing causally to the development of a specific condition of interest when alterated. Subsequently, in addition to the ranking via the score, finite Bayesian mixture models were developed and employed to classify

the transcripts w.r.t. being a 'driver' or not. Specifically, mixtures involving both normal distributions as well as exponential (or gamma) distributions were considered besides common mixtures of several normal distributions. To the author's best knowledge, when other authors used the difference of values from an omics variable measured for two collectives as base input in a mixture model before, only normal distributions were mixed and only one omics variable was analyzed (such as in, e.g., Wu and Ji, 2010).

First, the congruence of ChIP-chip and ChIP-seq technologies was assessed in Section 5.3 by a normal mixture model with eight components. Histone modifications in a murine cell line into which the oncogene BCR-ABL was transduced were investigated in comparison to a normal reference cell line. The mixture weights of the model components that represent either concordant or discordant differences add up to around only 20%, while the deviation of most values from zero is estimated to be so small that it may be explained by random variability. One reason for this may be that the same antibody (anti-H3K9Ac/K14Ac) was used for both the BCR-ABL and reference cell line, possibly leading to overly similar ChIP-seq profiles in them.

A proportion of 7.4% of transcripts was classified as presenting concordant differences, while a smaller fraction was classified as showing discordant differences (5.3%). According to the analysis, it can thus be concluded that ChIP-chip and ChIP-seq technologies agree more often than they disagree. Nevertheless, the considerable amount of discordances remains disturbing to practical researchers needing to rely on the validity of results. There are several reasons that potentially limit the generalizability of the results regarding the amount of discordances. First, the reference cell line specimens were originally treated differently for ChIP-seq (with a growth factor) and for ChIP-chip (without a growth factor). To increase comparability, several months later new ChIP-chip profiles were generated with a growth factor, but the considerable time lying between the two measurements may possibly have led to batch effects. In addition, further unwanted side effects of the growth factor cannot be ruled out.

Moreover, the sequencing depth in this study is relatively small, so a number of low- and medium-size peaks in the sequencing data may have gone undetected. Finally, the quality of the ChIP technology in general and the cross-platform validity of ChIP-chip and ChIP-seq measurements in particular depend heavily on the selection of an appropriate antibody for a specific histone modification. In a study carried out by Ho *et al.* (2011), an antibody similar to the one used in the experiments discussed here led to the lowest ChIP-chip/ChIP-seq correlation across several antibodies. For these reasons, the experimental conditions may have harmed the results of the analysis.

In Section 5.4, sequencing-based ChIP-seq measurements and gene transcription measurements (stemming from either microarrays or sequencing experiments) were analyzed integratively on several experimental and one simulated data set. For this purpose, a mixture model employing normal and exponential or gamma distributions was developed and applied. In the experimental data sets focusing on activating histone modifications that facilitate gene transcription, roughly between 1% and 10% of transcripts were classified as drivers, i.e. this proportion of transcripts showed deviations between case and reference samples that are equally directed in both ChIP and transcription data. Only roughly 0.1% to 1% of transcripts were classified as presenting deviations in opposite directions, while the rest was classified to neither of these two cases.

Regarding the association between gene transcription and histone modifications, the overall results are consistent with the literature (Zhang *et al.*, 2009). Findings from complimentary KEGG pathway analyses underscore their plausibility, and genes known for their association with the investigated phenotypical condition are identified. A simulation study as well as the comparison of two biological replicates from a data set based on research on the *Cebpa* gene indicate that the approach achieves a high specificity when detecting potential driver transcripts. Furthermore, the simulation study gives evidence for a reasonable sensitivity. A good reproducibility of the classification results is confirmed when analyzing the *Cebpa* data set after splitting it into two data sets of sample size

one.

When fitting the mixture model with the EM algorithm, similar to work presented in Taslim *et al.* (2009), the results were sometimes dependent on the set of initial values due to the algorithm's convergence into different local maxima. In the face of such a situation, it is common practice to select the results of the best fit, i.e. in this case, the fit leading to the highest likelihood. Disturbingly, however, some initial values did not result in reasonable classifications (i.e., classification leading to contiguous classes) although they did lead to a higher likelihood. When fitting a model via MCMC algorithms, on the other hand, it was necessary to specify informative prior distributions in order to ensure a reasonable classification. One might thus try to make the comparison fairer by using a likelihood with a penalty term aiming to enforce contiguous clusters as criterion for the EM fit. Regardless of this aspect, the MCMC framework facilitates to extend the model in several ways, e.g., interactions and spatial correlation between the transcripts may be taken into account and modeled as well. The EM algorithm, on the contrary, does not allow for such extensions.

Compared to a naive analysis assessing transcripts separately for differential transcription and histone modification based on fold changes, the presented Bayesian approach demonstrated to perform better in the simulation study. This is probably owed to the fact that the classification obtained from the Bayesian model is already based on aggregated information from both variables. In case of the naive approach, however, classification is first done separately for each data type, losing information. The findings underscore the need for novel integrative methods that can be applied in studies involving more than one omics data type. While model-based classification may consume more computing time, it avoids the need of choosing arbitrary thresholds inherent to naive approaches based on p-value or fold change rankings.

The presented Bayesian mixture model successfully mixes distributions of different types to achieve both a good fit and a reasonable classification to three contiguous classes. In preliminary analyses, a mixture model employing ex-

clusively normal components needed 15 components in order to achieve both goals. The proposed new model only needs six, requiring only one component for the classes corresponding to concordant and discordant differences, respectively. Thus, by mixing a fixed number of exponential or gamma distributions with normal ones, the necessary number of components can be kept small. This, while less flexible in terms of fit, is beneficial in terms of interpretability. It may often be preferable when classification is the major goal of the analysis, in particular when the number of clusters is obvious based on the underlying biological question. For instance, Kormaksson *et al.* (2012) employ a two-component normal mixture model to classify probe sets as being low or high methylated. Broet and Richardson (2006) as well as Wei and Pan (2008) employ three-component models to classify loci as being subject to loss or gain of genetic material, or else being unmodified. The small fixed number of components furthermore helps to avoid label switching problems, frequent in mixture models employing a flexible large number of normal components. However, although the modeling approach presented here was validated on several different data sets, it cannot be ruled out that a different number of components might be optimal for data sets possessing a different structure. The proposed framework is not specific for the application to gene transcription and histone modification data. It is seen as appropriate also for other classification tasks in the context of integrating two types of omics data when the sample size is small.

While fixing the number K of components has some advantages, as mentioned, one might still prefer to have it estimated by the model in order to avoid any arbitrariness in the choice of K. The added value of additional exponential (or gamma) components for the tails of the distribution is unclear in the presented modeling framework, since the higher flexibility provided by gamma distributions compared to exponential distributions already has drawbacks for the analysis' specificity and interpretability. However, one may argue that permitting a flexible number of normal components for the null class would be advantageous. The use of one exponential (or gamma) component for the

negative and one for the positive z_i values, respectively, does most of the work to achieve a low number of components and is essential in terms of classification and interpretation. In addition, the number of normal components necessary to represent the center of the Z distribution will always tend to be relatively small. Finally, the normal components are only interpreted as a whole, jointly represent the group of transcripts with z_i values equal to or near 0. For this reason, label switching between the normal components is not harmful. Also, a flexible number of such components could lead to a better fit in some cases and would largely exempt the user from the potential need of choosing the right number of normal components when dealing with structurally different data. A wealth of literature on models with a flexible or infinite number of components is available. For instance, the mixture of the normal components could be based on the finite approximation to a DPM model as described by Ishwaran and Zarepour (2000) and Ishwaran and James (2001). Fixing the number of exponential or gamma components, one would not have to worry about overfitting as much as in a standard mixture model with a flexible number of components (Frühwirth-Schnatter, 2011). Finally, other distributions than those considered here might also be incorporated in the mixture.

Instead of fitting a mixture distribution, one might also consider to approximate the distribution of each of the two factors that are multiplied to calculate the Z score by a normal distribution. This strategy would cause Z to have a normal-product distribution. Presumably, this would work well for data sets leading to roughly equally formed positive and negative tails in the Z distribution, when the main goal is to achieve a good fit of the distribution. The model presented in this thesis is probably more appropriate for classification, representing each class by one or more distinct mixture components and moreover allowing one tail of the Z distribution to contain considerably more probability mass than the other one.

When larger sample sizes are available, the externally centered correlation coefficient introduced by Schäfer *et al.* (2009) could be used as a modified Z

measure. However, instead of summarizing across the samples and fitting a mixture model to the resulting values, it may be even more promising to analyze the summands of the coefficient (used in Schäfer *et al.*, 2009, to construct a Wilcoxon rank sum test) in a Bayesian framework by assigning them a prior distribution across samples, e.g., a normal distribution. In this way, uncertainty could easily be modeled also across samples for each transcript.

However, assessment of uncertainty on a transcript level is even possible with a small number of samples. In this case, either the need for information has to be reduced by restricting the parameter space or additional information has to be borrowed from transcripts presenting a similar behavior. Both strategies have been explored already in works analysing single omics inputs, e.g., by Robinson *et al.* (2010) or Anders and Huber (2010) in the context of investigating differential gene transcription. A promising path aiming to incorporate additional information would be to employ pathway information about functionally related genes in order to define between which transcripts information should be shared.

One appropriate strategy to do this could be to employ a conditionally autoregressive (CAR) (or Markov random field) prior. It is frequently employed in, e.g., spatial epidemiology, defining adjacent regions as neighbors, and complemented by a spatially unstructured component in a so-called convolution model (Besag *et al.*, 1991). While in epidemiology, disease cases are in focus for which a Poisson distribution is often assumed, convolution models have also been applied to continuous variables of interest, using a normal distribution instead (Fahrmeir and Lang, 2000; Kandala *et al.*, 2001). To identify functionally related genes, several gene networks have been published which incorporate knowledge from a range of previous works, see, e.g., Lee *et al.* (2011) for humans or Guan *et al.* (2008) for mice. The functional similarity in such networks is often represented not only by neighborhood definitions, but by a quantification of the similarity of transcripts. Such similarity measures might readily be used as weights in the definition of an intrinsic CAR prior, where the conditional

mean given the neighbors may be specified not only as a simple, but also as a weighted mean across the neighbors. Wei and Pan (2008), e.g., successfully employed a CAR prior to integrate gene transcription and ChIP-chip transcription factor data, although not to achieve modeling on a transcript level, but to incorporate additional information in a mixture model similar to the ones presented in this thesis. Hu *et al.* (2011) use a CAR prior to represent correlation between base-specific RNA-seq signals, and Chen (2011) employ a CAR prior to take pathway information into account when assessing the impact of genetic variants on diseases in genome wide association studies.

Another challenge is the integration of more than two omics data types. For instance, it might be of interest to integrate DNA copy number data in addition to gene transcription and histone modification, if available. Another focus might be to jointly analyze several histone modifications. By modifying the Z coefficient in a way that allows to measure congruent aberrations for three or more variables, the subsequent model would remain independent from the number of integrated variables. It could thus be specified in the same way as in the case of two dimensions. The interpretation of a three-way correlation measure is not straightforward, however, and would need to be given special attention.

Appendix A

Additional Documentation and Results

A.1 Technological Background of Omics Measurements

Microarrays allow to measure, e.g., the transcription level, the DNA copy number or the level of DNA methylation simultaneously at locations covering the entire genome. An example is the Affymetrix GeneChip, consisting essentially of a quadratic object slide (sometimes referred to as 'array' or 'chip') with a side length of 1.28 cm. All microarray brands involved in this PhD thesis contain a grid of at least several hundreds of thousands of cells, the *probe cells*. Each probe cell in turn contains millions of a specific *oligonucleotide probe* (short: *oligo*). The oligos are fixed on the chip surface in different ways. In Illumina BeadChips, e.g., the oligos are first attached to silica beads, which are in turn randomly deposited into wells on the chip surface.

Oligos are short sequences containing 25 bases of DNA from a (normally artificial) reference genome. In principle, they can be unambiguously assigned to a specific gene, i.e. no other gene should contain the same sequence. In the microarrays used in the context of this thesis, all oligos are *perfect match oligos*,

i.e. exact complements to a DNA sequence of interest.

Both humans and mice have more than 20 000 genes. On a gene expression microarray, generally each of these genes is represented by several probes, which may be grouped to *probe sets* consisting of several probes each. The strategy by which probes are distributed along the genome varies between array types and manufacturers. Potentially, there are also probe sets that only serve for quality control, without representing a gene. Depending on the array design, a probe can measure several transcripts with potentially different TSSs. *Tiling arrays*, a specific type of microarrays, are designed to represent the genome at a particular high resolution of between 100 and a few base pairs and generally contain millions of probes. On chips designed to study protein/DNA interactions, probes may concentrate on the promoter regions since many histone modifications occur primarily near the TSSs.

To obtain measurements from a microarray, first single-stranded DNA (for investigating, e.g., histone modifications or DNA copy number) or RNA (for investigating gene transcription) of an individual is applied to the array. This so-called *target DNA* or *target RNA* subsequently *hybridizes*, i.e. the fragments bond with their complimentary sequences fixed on the array surface. Fragmenting the targets to a length of about 20 to 100bp in a prior step may increase efficiency and specificity of the hybridization. Normally, complementary DNA or RNA (short cDNA or cRNA) recovered from mRNA is used because it does not contain non-coding introns. Prior to hybridizing, it is marked with fluorescent dye and copied multiple times by the means of polymerase chain reaction (PCR). This is necessary to generate enough genomic material in order to obtain light signals with a sufficient intensity. After hybridization, the material not bound to the array is removed by a washing step. The array is then optically scanned to measure the intensity of the fluorescent dye for each probe. Each probe cell is captured by a number of pixels whose intensity values are aggregated to a single value per probe cell. The crucial assumption underlying the downstream analysis is that the intensity of the dye is proportional to the

amount of the hybridized specific DNA or mRNA.

The mapping between probes and transcripts is complicated by *cross-hybridization*, i.e. the fact that sometimes mRNA does not hybridize to its corresponding probe, but a probe representing a very similar sequence. Typically, this may happen in the case of genes sharing a common ancestor. Cross-hybridization and imprecisions resulting from, e.g., the optical system, printing during microarray production, hybridization or scanning of the fluorescence level may contribute to background noise in the signal. Such noise affects the measurement quality and leads to a bias in signal intensities. Since this in turn may result in the false discovery of differences between signals, probe-level intensities have to be corrected for background noise and normalized to remove unwanted effects. The specific approach employed in this thesis is discussed in Sections 5.3.2 and 5.4.2.

While microarrays allow a high throughput at often relatively low costs, they also have several drawbacks (Wang *et al.*, 2009). In particular, they only measure at predefined genomic loci taken into account in the array design and thus rely on prior knowledge about the genome. Also, as mentioned before, cross-hybridization between distinct loci may occur and lead to high background signals. Finally, the measurements are precise only in a certain signal interval: very low values tend to be masked by background noise and the proportionality assumption of transcription level and fluorescence intensity does not always hold. Specifically, it does generally not hold for high values due to saturation.

Motivated by these limitations, in the last years sequencing approaches have been developed that are able to directly determine cDNA sequences without relying on the in-between step of hybridization. They provide a much higher resolution and sensitivity, decreasing the risk of false negatives when searching for, e.g., differential gene transcription or rare transcripts (Marioni *et al.*, 2008; Mortazavi *et al.*, 2008). In this thesis, the focus lies on the RNA-seq and ChIP-seq technologies designed to measure gene transcription and DNA methylation, respectively.

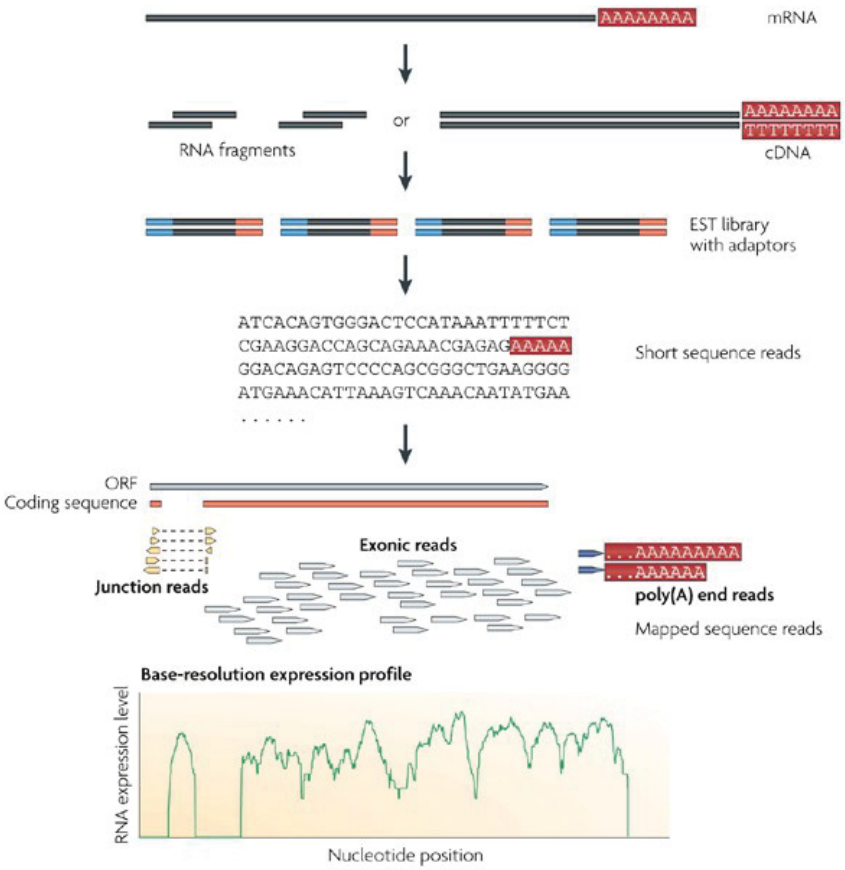

Figure A.1: Principal steps of an RNA-seq analysis. First, mRNAs pieces are converted into a library of shorter cDNA sequences via fragmentation. After sequencing adaptors (blue) are added to each cDNA fragment, a short sequence is obtained from each of the fragments using high-throughput sequencing technology. The resulting sequence reads are aligned to a reference genome, and classified as three types: exonic reads, junction reads and poly(A) end-reads. These three types are used to generate an expression profile for each transcript at base-resolution. Source: Wang *et al.* (2009).

The essential steps of an RNA-seq analysis are visualized in Figure A.1. In a population of mRNA fragments to be analyzed, at first large pieces need to be fragmented into smaller ones in order to achieve lengths of about 200–500 bp, compatible with most actual sequencing technologies. There are two standard fragmentation methods to accomplish this, RNA fragmentation and cDNA fragmentation. Both eventually result in the conversion of mRNA pieces to cDNA fragments. A sequencing adaptor is added to one or both ends of each of these cDNA fragments. Mostly, PCR is performed to obtained sufficient genomic ma-

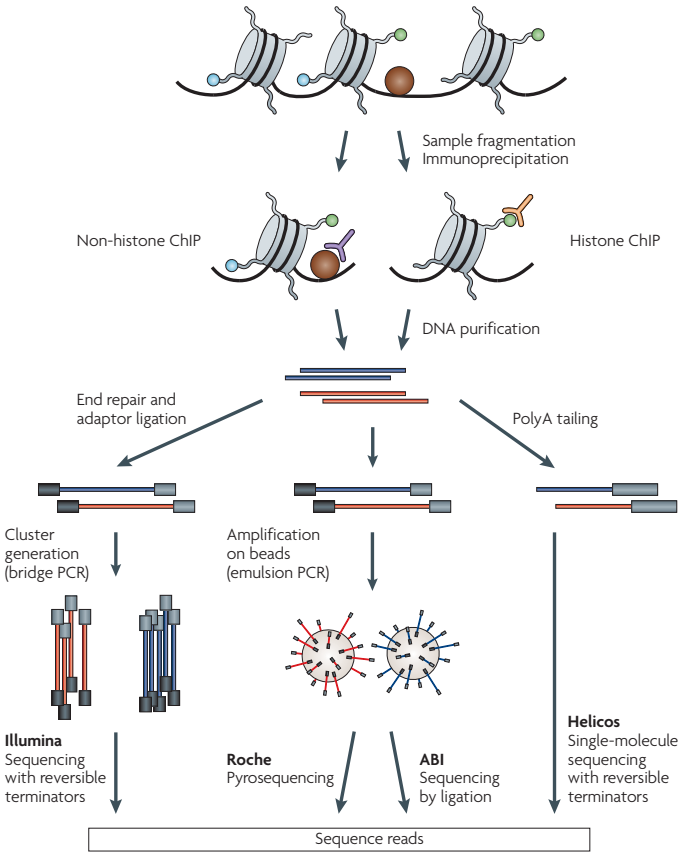

Figure A.2: Principal steps of a ChIP-seq analysis. First, the DNA-protein complexes are fragmented into shorter pieces. The process of chromatin immunoprecipitation (ChIP) enriches theses complexes of DNA and proteins of interest (e.g., histones) via an antibody specific to these proteins. After the DNA is purified and adaptors are added, sequencing starts on one of several available next-generation platforms, involving the enzyme-driven cloning of all templates in parallel (PCR amplification). After each extension, incorporated fluorescent labels are detected by high-resolution imaging. Source: Park (2009).

terial. Finally, the sequencing consists in producing a short sequence from each cDNA piece, either from one end (single-end sequencing) or both ends (pair-end sequencing) of each fragment. The size of these short reads may be 30–400 bp, although it is typically smaller than 100bp, depending on the employed platform (such as Illumina, Roche, ABI or Helicos).

To determine their origin, the reads resulting from sequencing are aligned to a (mostly artificial) reference genome. Apart from exonic sequences, the focus in this thesis, reads may also correspond to exon junctions or poly(A) ends that

only have adenine bases (fulfilling a special function during translation).

The principal structure of a ChIP-seq analysis has similarities to the one of RNA-seq in some steps and is summarized in Figure A.2. The DNA (together with the proteins attached to it) is first fragmented into smaller pieces, usually in the order of 200–600 bp. After employment of a protein-specific antibody for immunoprecipitation, implying the enrichment of complexes involving DNA and specific histones, the DNA is purified and amended with adaptors. Subsequently, it is usually amplified by PCR in order to dispose of a sufficient amount of genomic material. Finally, sequencing is carried out on one of several available platforms, producing sequence reads. The ChIP procedure depends essentially on finding an antibody possessing sufficient specificity for the histone modification of interest, as a low specificity results in a low chromatin and protein recovery efficiency. The technical implementation of the sequencing step depends on the employed platform. For the data analyzed in this thesis, the Illumina Genome Analyzer IIx is mostly employed, able to generate 100–400 million reads in a single run. In Figure A.3, the work flow of the initial version of this platform for single-ended sequencing is outlined: adapter ligation steps are performed before DNA fragments are immobilized at one end on the solid surface of a flow cell, densely coated with adapters and their compliments. Hybridizing to the surface-bound complimentary adapter with its free end, the fragments then form 'bridge' molecules which are amplified by PCR. Several PCR cycles eventually result in clusters of about 1000 identical fragments each. Then, strands of nucleic acid called *primers* are annealed to the fragments, serving as starting point for the DNA polymerase enzyme (see next step). Now, in the crucial step, sequencing-by-synthesis is performed: a reaction mixture is added containing primers, DNA polymerase and four so-called reversible terminator nucleotides, each for one of the bases and labeled with a different fluorescent dye. After incorporation into the DNA strand by the DNA polymerase enzyme, the labeled nucleotide at the first base/position is identified by a camera via its specific fluorescent dye. The synthesis cycle is repeated until the sequence read

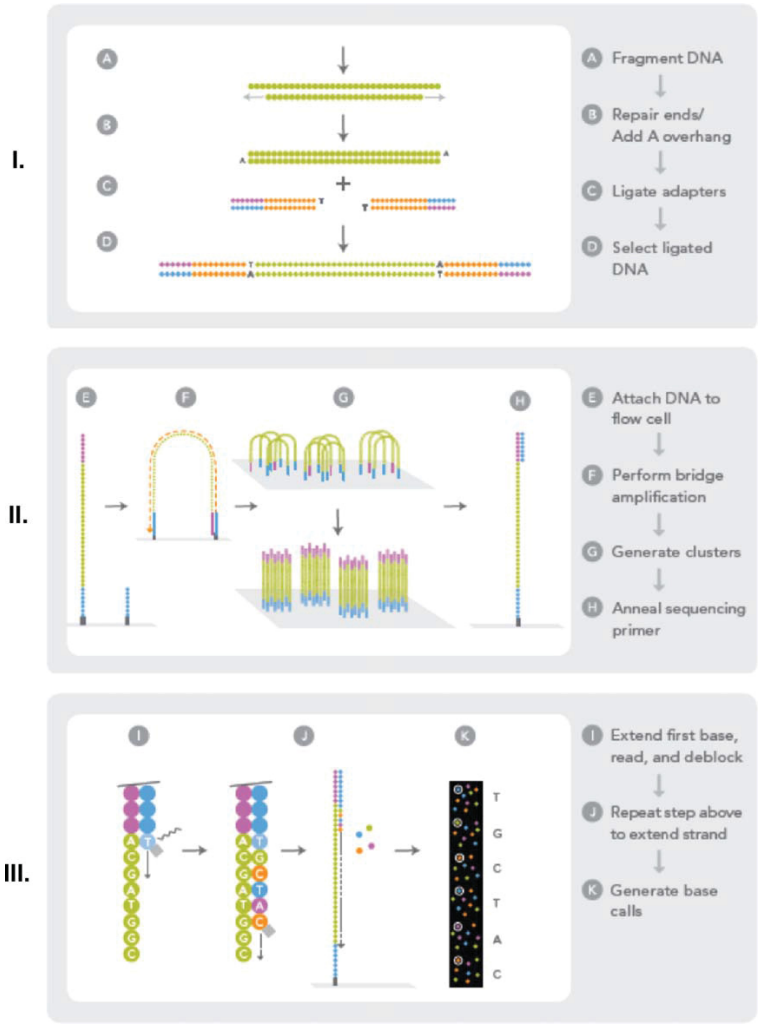

Figure A.3: Outline of the Illumina Genome Analyzer work flow. After adapter ligation, clusters of clonal sequences are generated by bridge PCR, and sequencing is performed by sequencing-by-synthesis: A reaction mixture containing primers and DNA polymerase is added as well as four reversible terminator nucleotides labeled with a different fluorescent dye, each for one of the bases. After each extension, the fluorescent labels that have been incorporated are detected through high-resolution imaging. Source: Ansorge (2009).

length is about 35 nucleotides. The sequence of 40 million clusters containing about 1000 identical fragments each can be determined in parallel, taking about two days for a single-end run and four days for a paired-end run on the Genome Analyzer IIx platform.

A detailed description of next-generation sequencing techniques from Illumina and other manufactors on a technical level can be found in Ansorge (2009).

A.2 GAMMICS Method: Full Conditional Distributions

Please also refer to Section 4.5 for notation and definitions used in the following. The joint likelihood is

$$p(\boldsymbol{D^2}, \boldsymbol{T}, \tilde{\alpha}_0, \tilde{\beta}_0, \tilde{\alpha}_1, \tilde{\beta}_1, p_c, a_{\tilde{\alpha}_0}, b_{\tilde{\alpha}_0}, a_{\tilde{\alpha}_1}, b_{\tilde{\alpha}_1}, c_{\tilde{\beta}_0}, d_{\tilde{\beta}_0}, c_{\tilde{\beta}_1}, d_{\tilde{\beta}_1}, a_{p_c}, b_{p_c}) \propto$$

$$p(\boldsymbol{D^2}|p_c, \tilde{\alpha}_0, \tilde{\beta}_0, \tilde{\alpha}_1, \tilde{\beta}_1) \cdot \prod_{k=0,1} p(\tilde{\alpha}_k|a_{\tilde{\alpha}_k}, b_{\tilde{\alpha}_k}) \cdot \prod_{k=0,1} p\left(\frac{1}{\tilde{\beta}_k}\middle| c_{\tilde{\beta}_k}, d_{\tilde{\beta}_k}\right)$$
$$\cdot \quad p(\boldsymbol{T}|p_c) \cdot p(p_c|a_{p_c}, b_{p_c})$$

where

$$\begin{aligned}
p(\boldsymbol{T}|p_c) &\propto \prod_{i=1}^{n} {p_c}^{T_i} \cdot (1-p_c)^{(1-T_i)}, \\
p(\boldsymbol{D^2}|p_c, \tilde{\alpha}_0, \tilde{\beta}_0, \tilde{\alpha}_1, \tilde{\beta}_1) &\propto \prod_{i=1}^{n} p(D_i^2|\tilde{\alpha}_0, \tilde{\beta}_0, \tilde{\alpha}_1, \tilde{\beta}_1, p_c) \\
&= \prod_{i=1}^{n} \left(p_c \cdot p(D_i^2|\tilde{\alpha}_1, \tilde{\beta}_1) + (1-p_c) \cdot p(D_i^2|\tilde{\alpha}_0, \tilde{\beta}_0)\right) \\
&= \prod_{i=1}^{n} \left(p_c \cdot \frac{1}{{\tilde{\beta}_1}^{\tilde{\alpha}_1}\Gamma(\tilde{\alpha}_1)} \left(D_i^2\right)^{\tilde{\alpha}_1 - 1} e^{-\frac{D_i^2}{\tilde{\beta}_1}}\right. \\
&\qquad \left. + (1-p_c) \cdot \frac{1}{{\tilde{\beta}_0}^{\tilde{\alpha}_0}\Gamma(\tilde{\alpha}_0)} \left(D_i^2\right)^{\tilde{\alpha}_0 - 1} e^{-\frac{D_i^2}{\tilde{\beta}_0}}\right),
\end{aligned}$$

and

$$\begin{aligned}
p(\tilde{\alpha}_k|a_{\tilde{\alpha}_k}, b_{\tilde{\alpha}_k}) &\propto \frac{1}{{b_{\tilde{\alpha}_k}}^{a_{\tilde{\alpha}_k}}\Gamma(a_{\tilde{\alpha}_k})} \alpha_y^{a_{\tilde{\alpha}_k}-1} e^{-\frac{\tilde{\alpha}_k}{b_{\tilde{\alpha}_k}}}, k = 0, 1, \\
p\left(\frac{1}{\tilde{\beta}_k}\middle| c_{\tilde{\alpha}_k}, d_{\tilde{\alpha}_k}\right) &\propto \frac{1}{{d_{\tilde{\alpha}_k}}^{c_{\tilde{\alpha}_k}}\Gamma(c_{\tilde{\alpha}_k})} \left(\frac{1}{\tilde{\beta}_k}\right)^{c_{\tilde{\alpha}_k}-1} e^{-\frac{1}{\tilde{\beta}_k d_{\tilde{\alpha}_k}}}, k = 0, 1, \\
p(p_c|a_{p_c}, b_{p_c}) &\propto \frac{1}{B(a_{p_c}, b_{p_c})} {p_c}^{a_{p_c}-1} (1-p_c)^{b_{p_c}-1},
\end{aligned}$$

where $B(\alpha, \beta)$ is the Beta function.

A.3 GAMMICS Method: Sampling Scheme

The full sampling scheme for the Gibbs Sampler is as follows:

- Resample the allocation (classification) indices $t_1, \ldots, t_n$ independently with probabilities

$$P(T_i = k|D_i^2, p_c) \propto \begin{cases} p_c & \cdot \quad \text{Gamma}(D_i^2|\tilde{\alpha}_1, \tilde{\beta}_1) \quad \text{for} \quad k = 1, \\ (1 - p_c) & \cdot \quad \text{Gamma}(D_i^2|\tilde{\alpha}_0, \tilde{\beta}_0) \quad \text{for} \quad k = 0. \end{cases}$$

- Update $a_{p_c} = a_{p_c,0} + \sum_i^n t_i$ and $b_{p_c} = b_{p_c,0} + n - \sum_i^n t_i$, then sample p_c from Beta(a_{p_c}, b_{p_c}), where $a_{p_c,0}$ and $b_{p_c,0}$ are the prior choices for a_{p_c} and b_{p_c}.

- For $k = 0, 1$, update $c_{\tilde{\beta}_k} = \tilde{\alpha}_k \cdot n_k + c_{\tilde{\beta}_k,0}$ and $d_{\tilde{\beta}_k} = \frac{d_{\tilde{\beta}_k,0}}{1+d_{\tilde{\beta}_k,0} \cdot \sum_{i:t_i=k} d_i^2}$, where $c_{\tilde{\beta}_k,0}$ and $d_{\tilde{\beta}_k,0}$ are the prior choices for $c_{\tilde{\beta}_k}$ and $d_{\tilde{\beta}_k,0}$. Then draw $1/\tilde{\beta}_k$ from Gamma$(c_{\tilde{\beta}_k}, d_{\tilde{\beta}_k})$.

- For $k = 0, 1$, independently draw $\tilde{\alpha}_k$ from Gamma$(a_{\tilde{\alpha}_k}, b_{\tilde{\alpha}_k})$ via a Metropolis-Hastings step, where the proposal distribution is a normal distribution centered at the current value and mirrored at 0 with standard deviation σ_{MH}.

- Calculate $\tilde{d}_c$ and estimate the cluster partition according to Algorithm 5.

- Based on the reconstructed cluster structure, calculate the corresponding sizes and radii for all resulting clusters. The radii are calculated as the mean pairwise distance between points within the cluster, multiplied with the factor $2 \cdot \frac{45\pi}{128}$ of the maximum distance between two points within the same cluster. For these calculations, a cluster is required to contain at least two points by definition.

- For a subsequent calculation of an estimate for the proportion of clustered proteins p_c, in each iteration the actual p_c value is saved, corrected to take

into account outliers in the distribution of the d_i^2 as non-clustered points. For a subsequent calculation of estimates for the mean cluster size μ_c and the mean cluster radius r_c, the mean size and the mean radius are saved for all clusters.

A.4 GAMMICS Method: Additional Results

A.4.1 Trace Plots

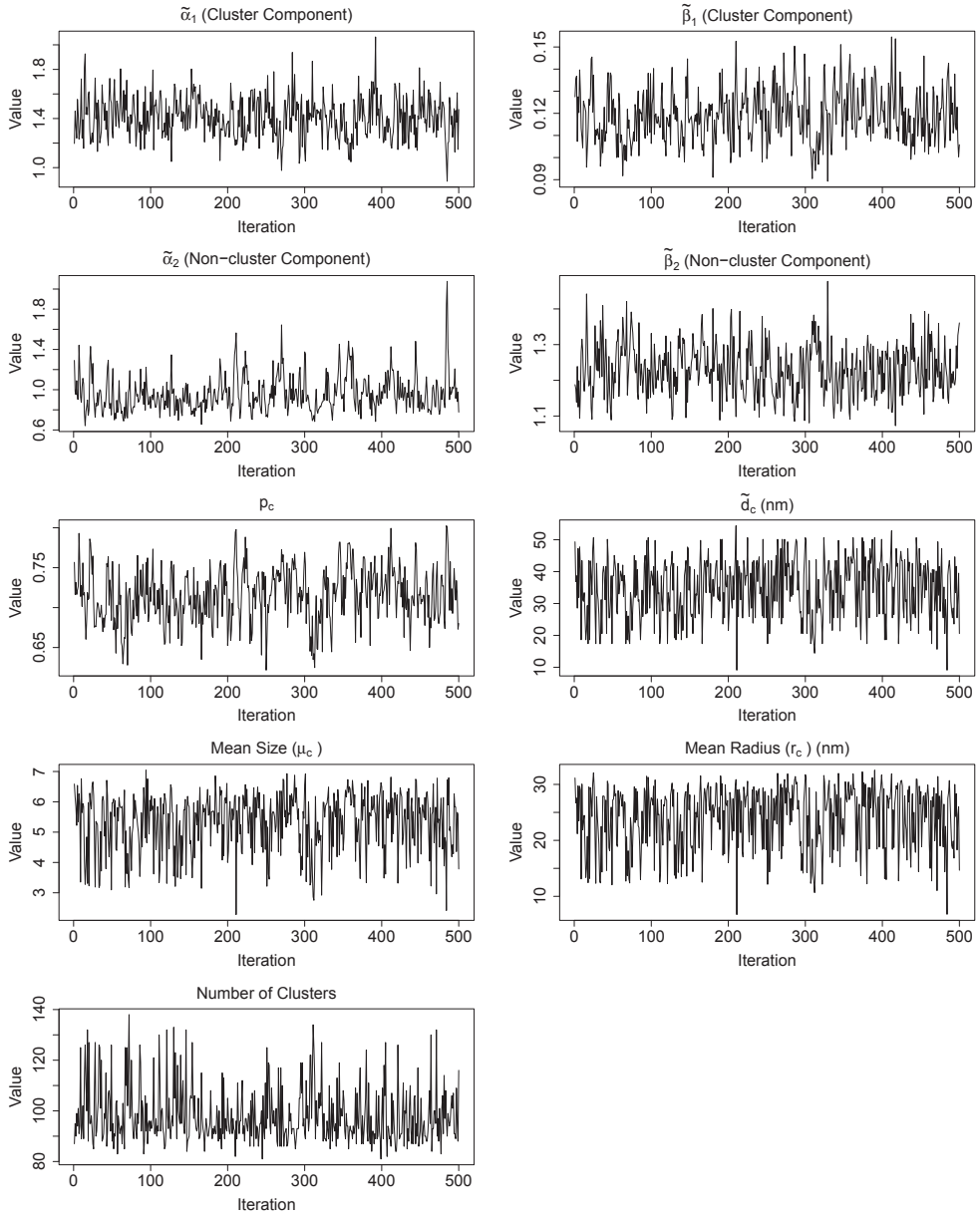

Figure A.4: Trace plots of the model parameters after fitting the GAMMICS method (see Section 4.5) to ROI 3 of the experimental data set as shown in Figure 4.2A. The first 8 000 iterations were discarded and the remaining iterations were thinned by keeping every 30th iteration. The remaining 500 iterations were used for parameter estimation and are shown in the trace plots.

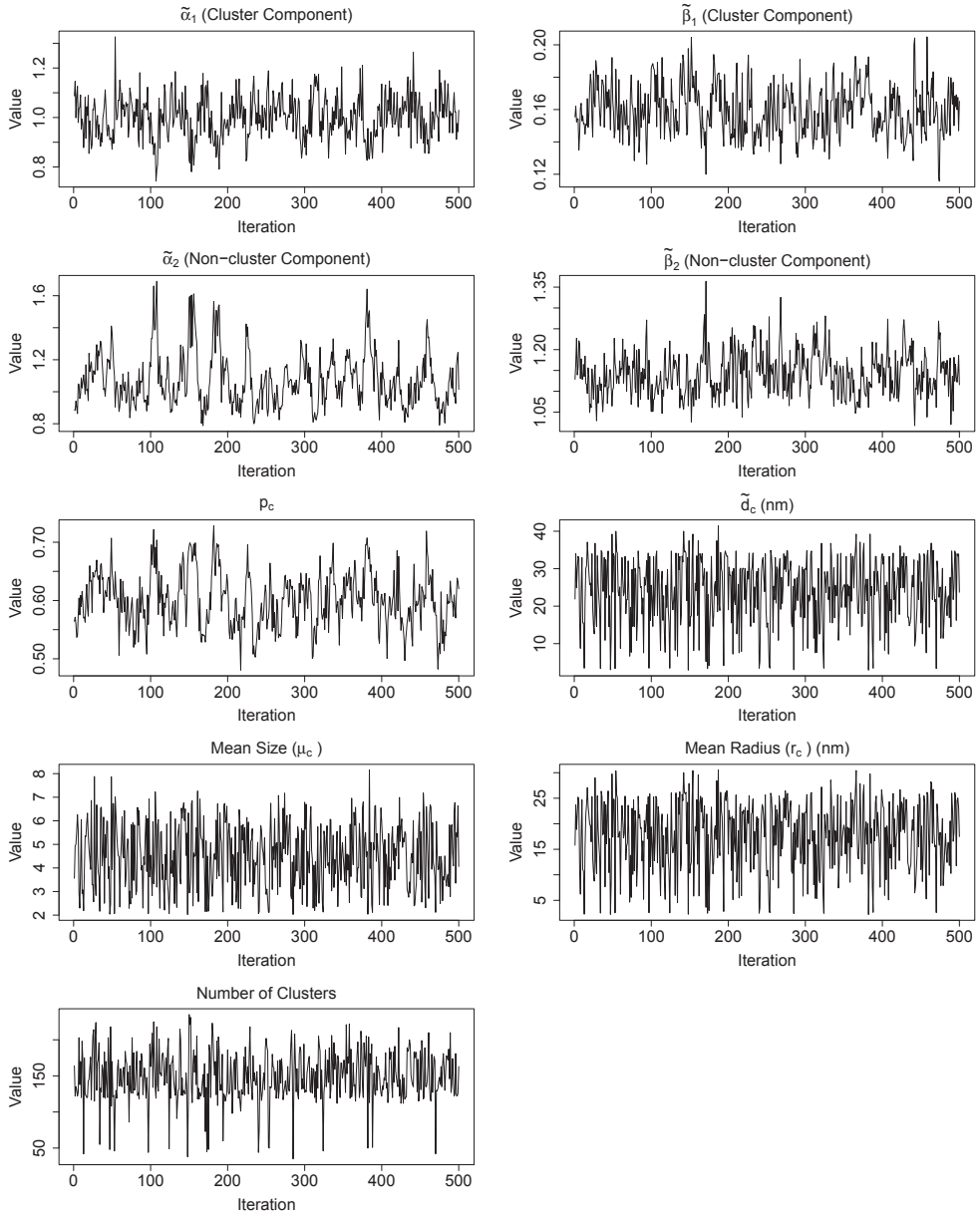

Figure A.5: Trace plots of the model parameters after fitting the GAMMICS method (see Section 4.5) to ROI 6 of the experimental data set. The first 8 000 iterations were discarded and the remaining iterations were thinned by keeping every 30th iteration. The remaining 500 iterations were used for parameter estimation and are shown in the trace plots.

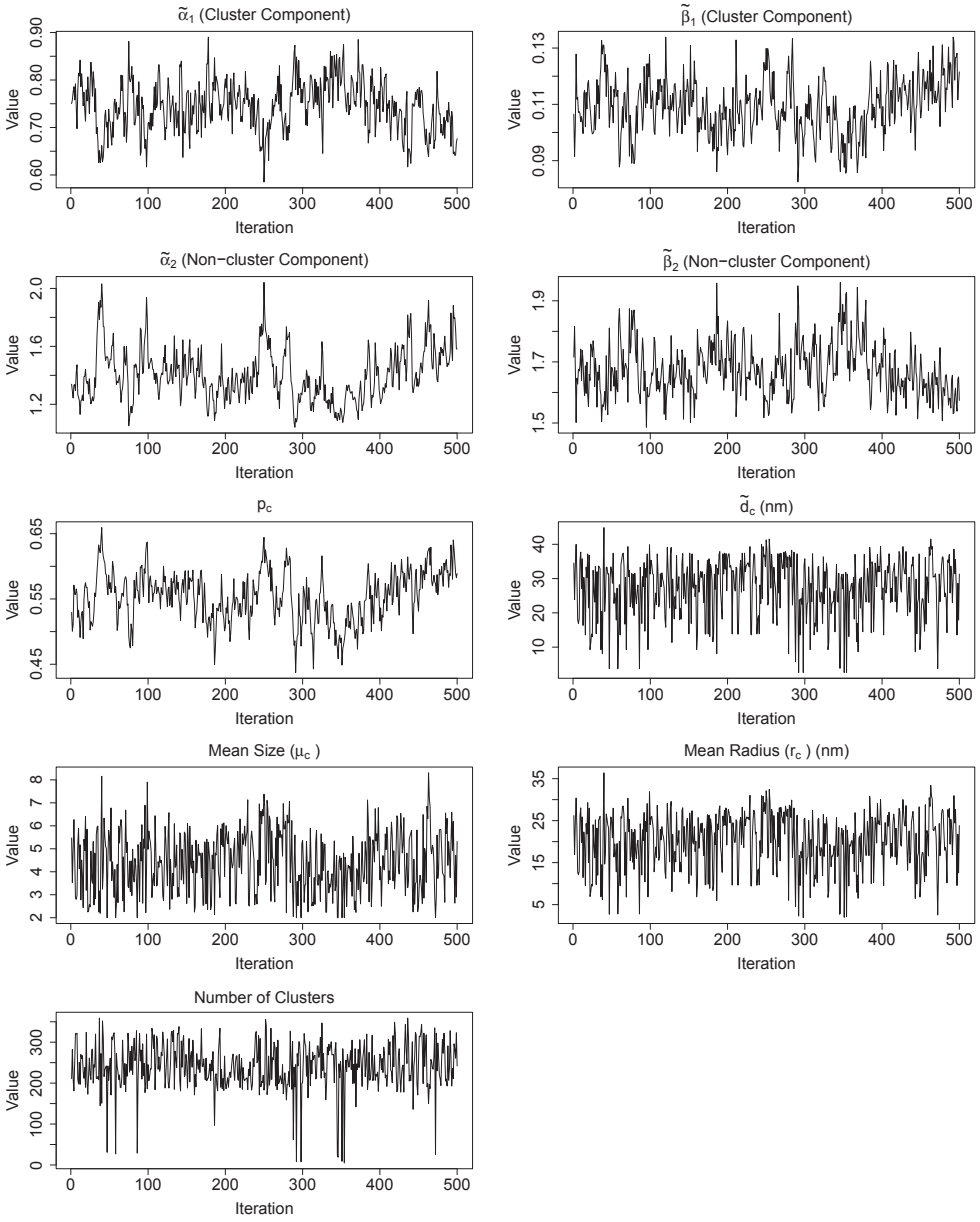

Figure A.6: Trace plots of the model parameters after fitting the GAMMICS method (see Section 4.5) to localizations simulated with a point density of 200 points/μm^2, proportion of proteins in clusters $p_c = 0.4$, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 15$nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, with a proportion of proteins in metaclusters of $p_{mc} = 0.6$. The first 8 000 iterations were discarded and the remaining iterations were thinned by keeping every 30th iteration. The remaining 500 iterations were used for parameter estimation and are shown in the trace plots.

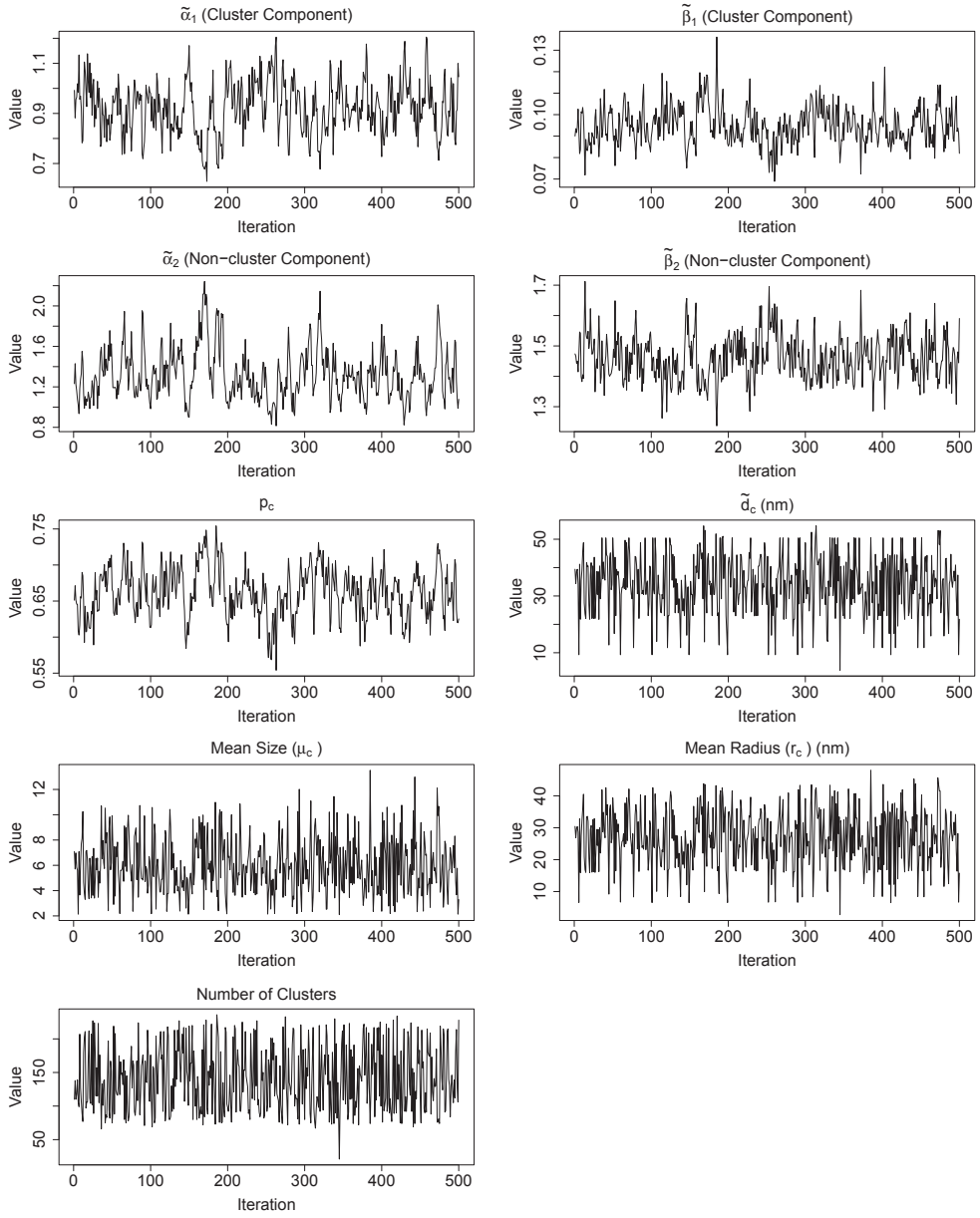

Figure A.7: Trace plots of the model parameters after fitting the GAMMICS method (see Section 4.5) to localizations simulated with a point density of 125 points/μm^2, proportion of proteins in clusters $p_c = 0.4$, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 30$ nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, with a proportion of proteins in metaclusters of $p_{mc} = 0.6$. The first 8 000 iterations were discarded and the remaining iterations were thinned by keeping every 30th iteration. The remaining 500 iterations were used for parameter estimation and are shown in the trace plots.

A.4.2 Posterior Histograms Across Clusters

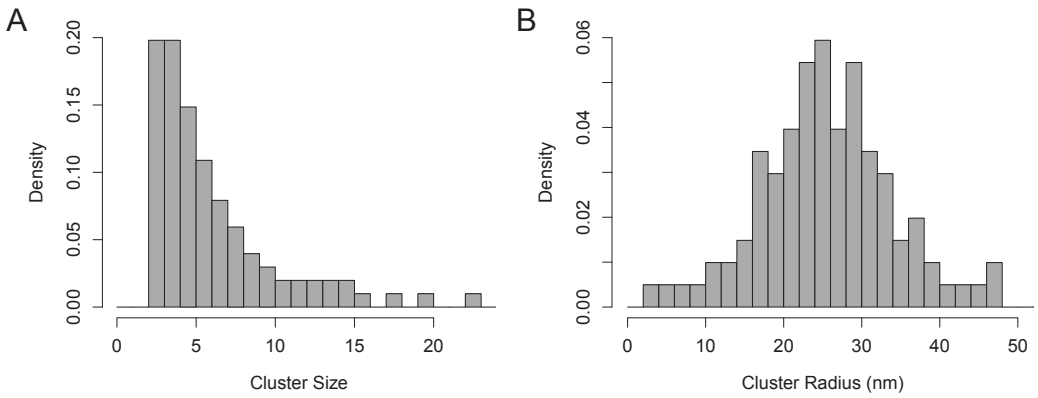

Figure A.8: Posterior histograms of (**A**) the cluster size and (**B**) the cluster radius for experimental data, ROI 3, based on 100 posterior quantiles of the distribution across the clusters (0%, 1%, 2%, ... quantiles).

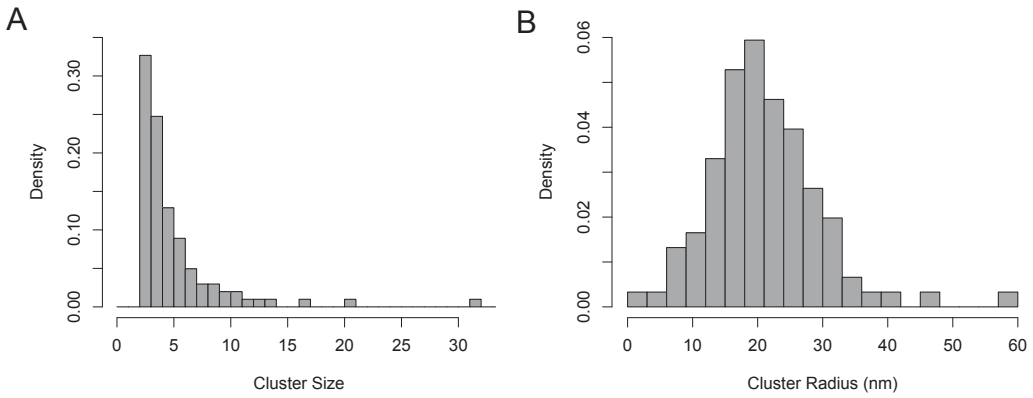

Figure A.9: Posterior histograms of (**A**) the cluster size and (**B**) cluster radius for localizations simulated with a point density of 200 points/μm^2, proportion of proteins in clusters $p_c = 0.4$, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 15$nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, with a proportion of proteins in metaclusters of $p_{mc} = 0.6$, based on 100 posterior quantiles of the distribution across the clusters (0%, 1%, 2%, ... quantiles).

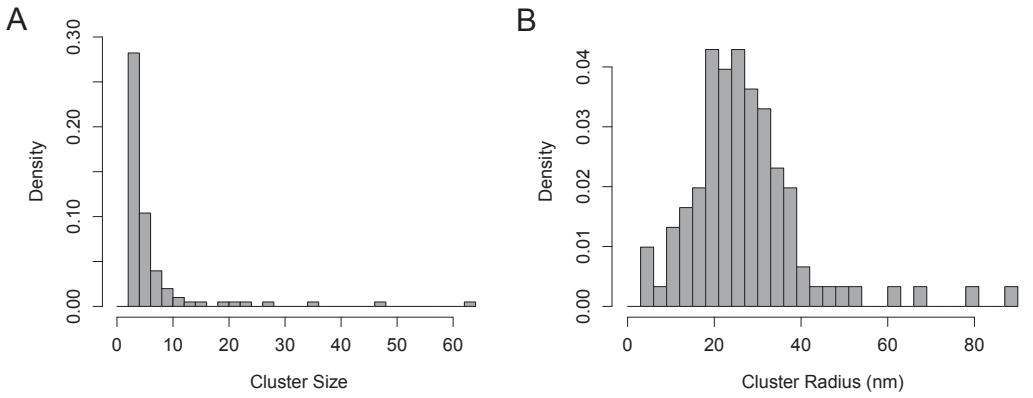

Figure A.10: Posterior histograms of (**A**) the cluster size and (**B**) cluster radius for localizations simulated with a point density of 125 points/μm^2, proportion of proteins in clusters $p_c = 0.4$, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 30$ nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, with a proportion of proteins in metaclusters of $p_{mc} = 0.6$, based on 100 posterior quantiles of the distribution across the clusters (0%, 1%, 2%, ... quantiles).

A.4.3 Posterior Histograms Across MCMC Iterations

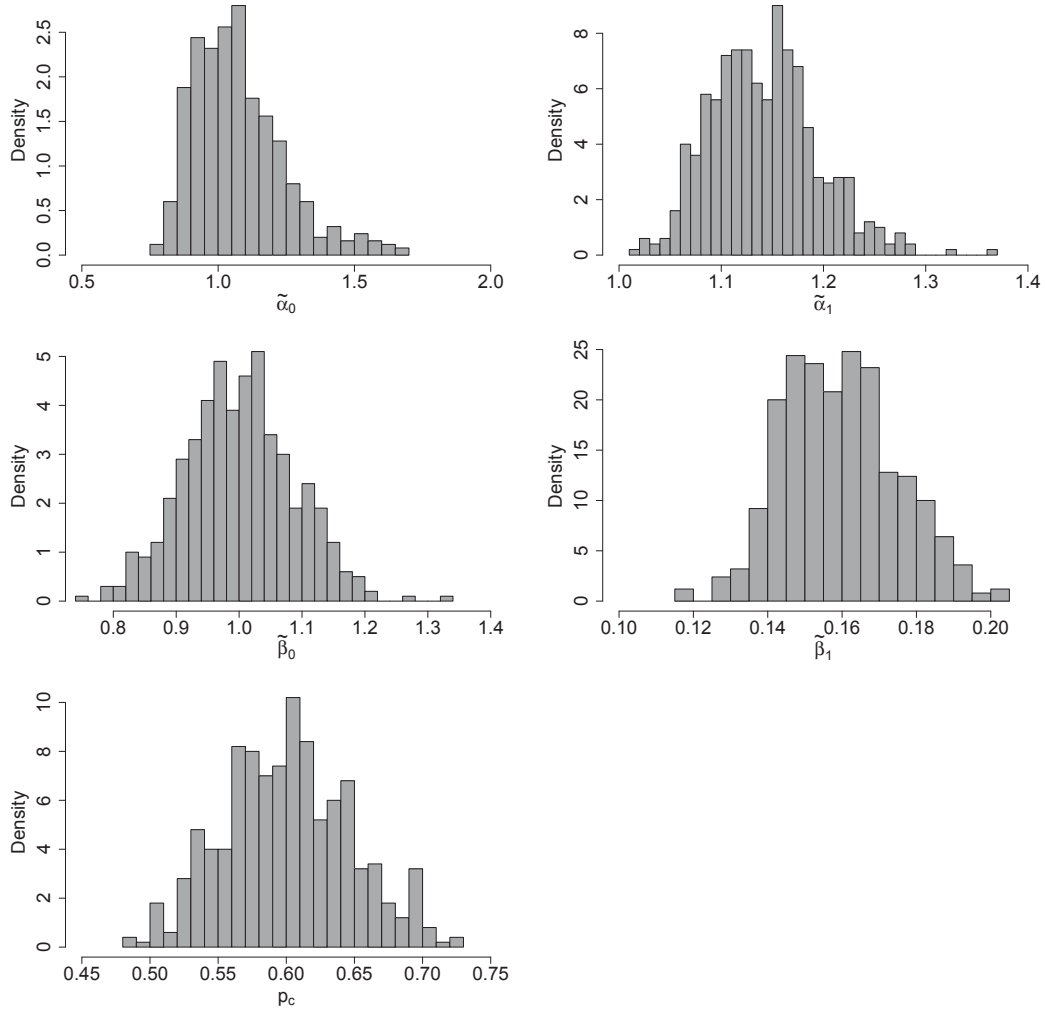

Figure A.11: Posterior histograms of model parameter distributions across MCMC iterations for experimental data, ROI 6. Shown are histograms for the parameters of the two gamma distributions as well as the proportion of clustered points p_c.

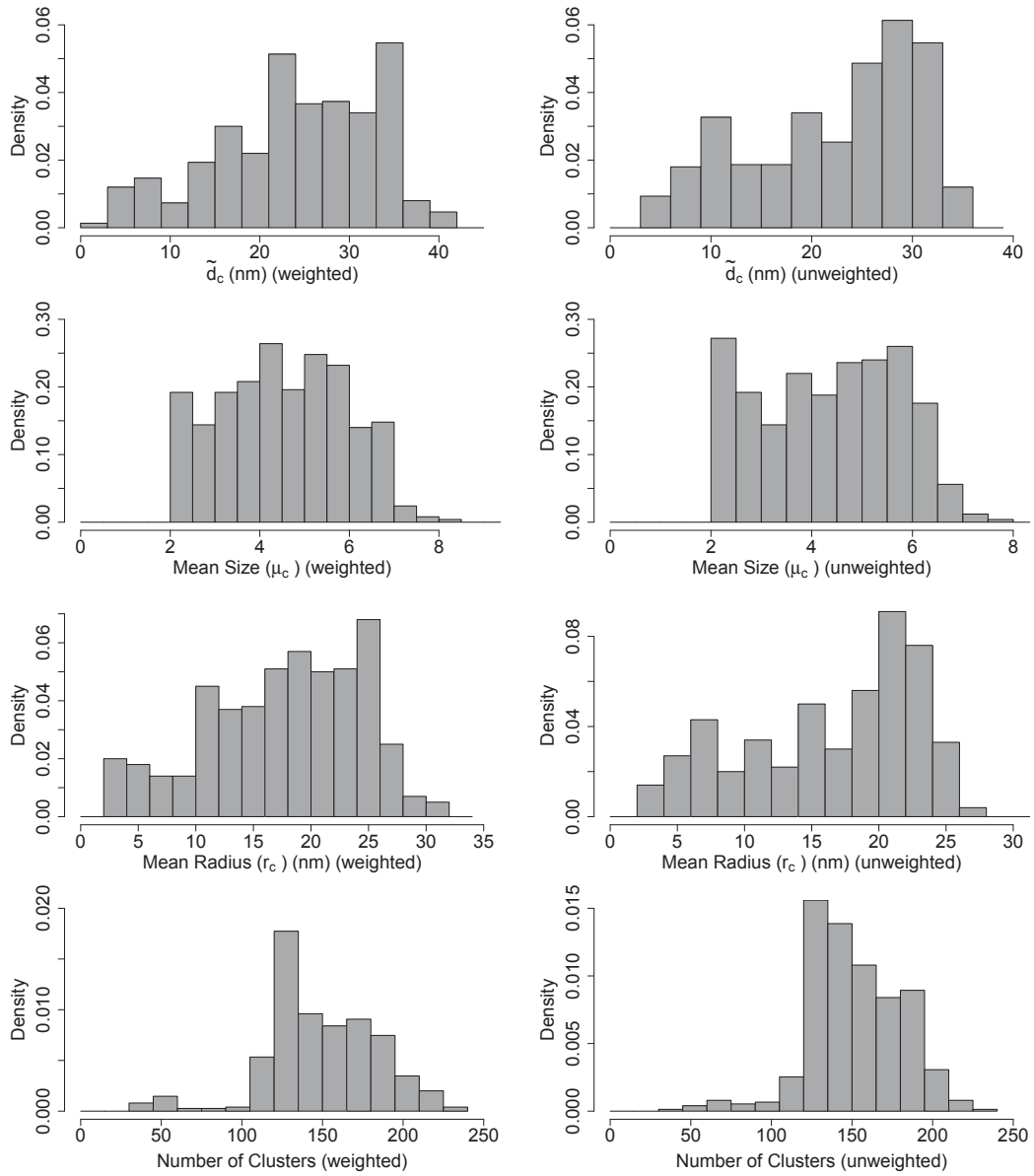

Figure A.12: Posterior histograms of model parameter distributions across MCMC iterations for experimental data, ROI 6. Shown are histograms for the cutoff $\tilde{d}_c$, the mean cluster size μ_c, the mean cluster radius r_c, and the number of clusters. Each histogram is shown twice, on the left hand side for the calculation of $\tilde{d}_c$ with weighted gamma distributions and on the right hand side or the calculation of $\tilde{d}_c$ with unweighted gamma distributions.

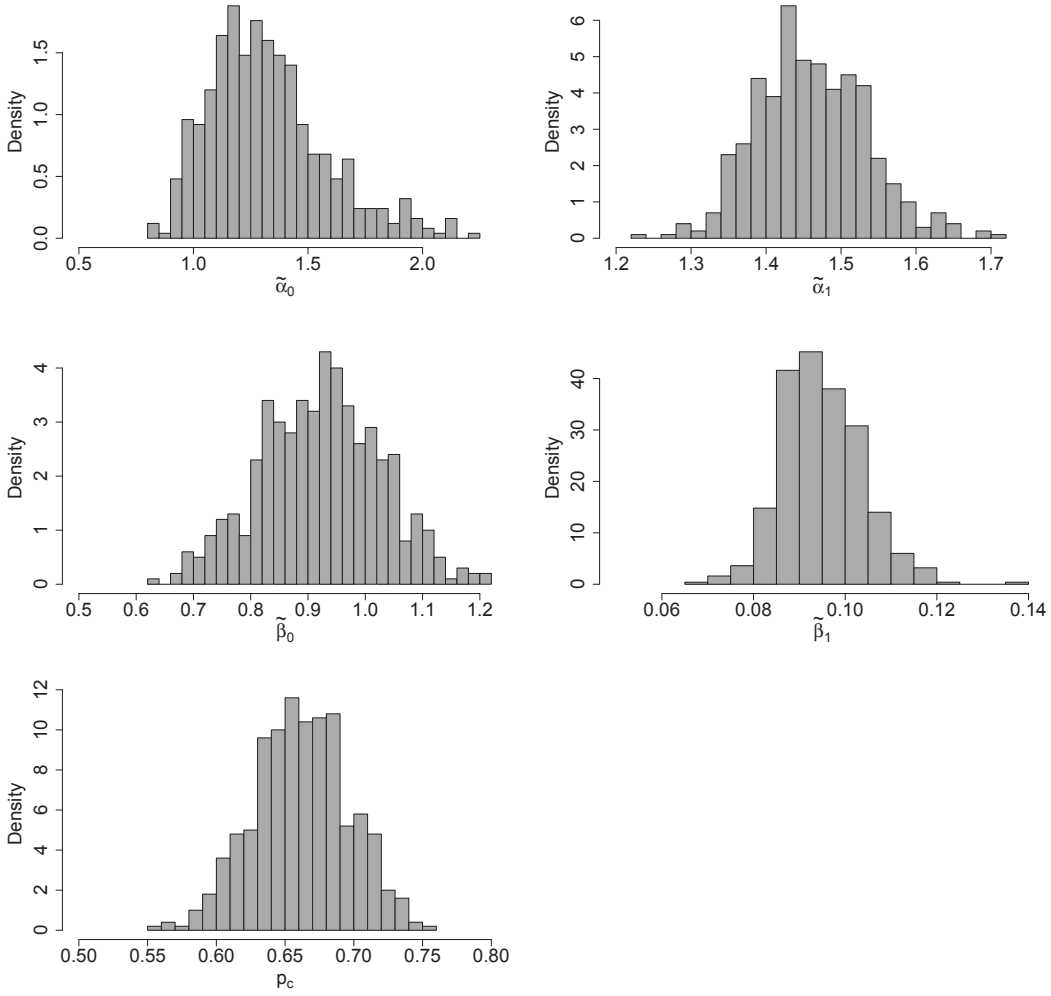

Figure A.13: Posterior histograms of model parameter distributions across MCMC iterations for localizations simulated with a point density of 125 points/μm^2, proportion of proteins in clusters $p_c = 0.4$, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 30$ nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, with a proportion of proteins in metaclusters of $p_{mc} = 0.6$. Shown are histograms for the parameters of the two gamma distributions as well as the proportion of clustered points p_c.

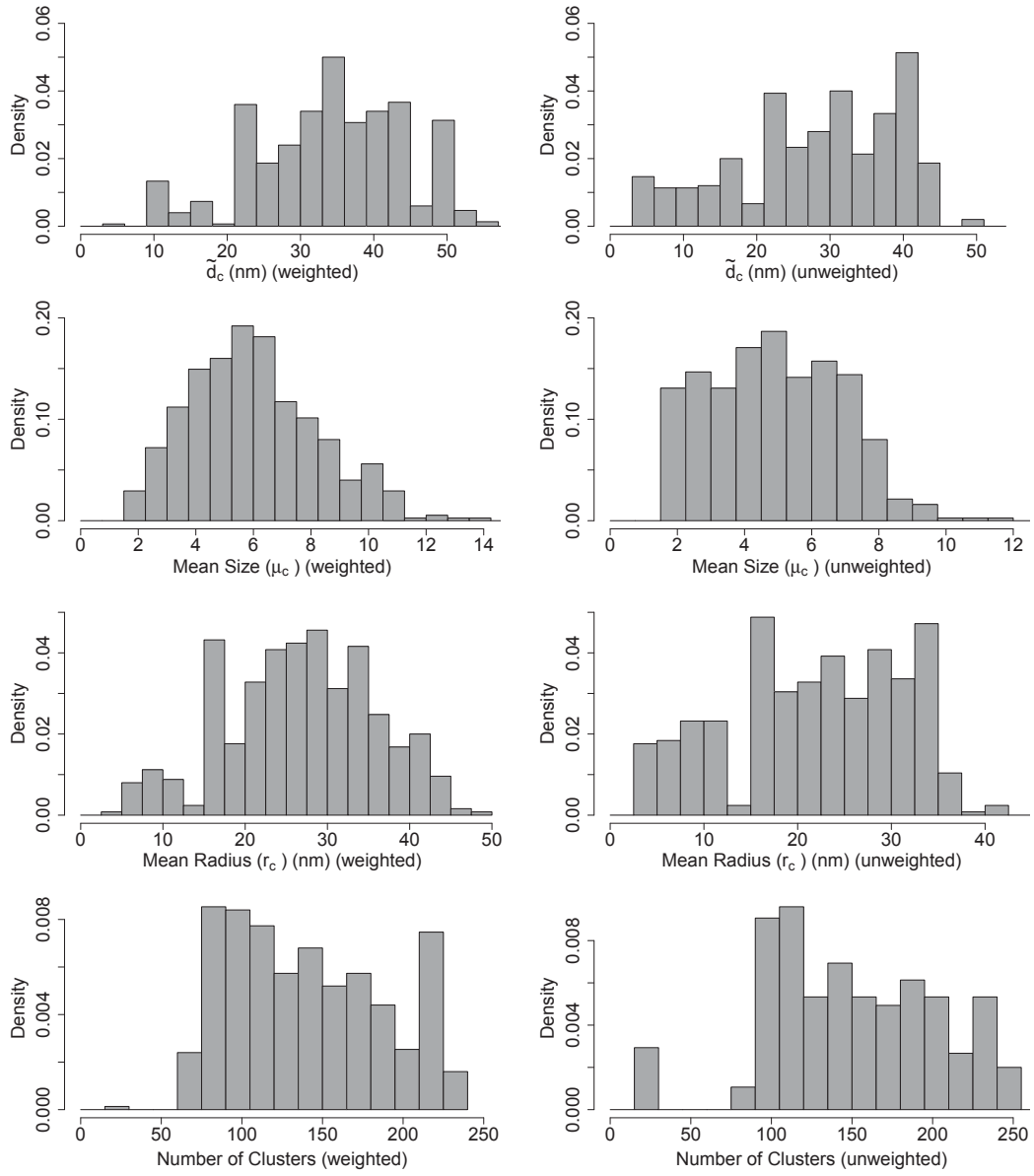

Figure A.14: Posterior histograms of clustering parameter distributions across MCMC iterations for localizations simulated with a point density of 125 points/μm^2, proportion of proteins in clusters $p_c = 0.4$, mean cluster size $\mu_c = 4$, mean cluster radius $r_c = 30$ nm, mean metacluster size $\mu_{mc} = 20$, mean metacluster radius $r_{mc} = 100$ nm, with a proportion of proteins in metaclusters of $p_{mc} = 0.6$. Shown are histograms for the cutoff $\tilde{d}_c$, the mean cluster size μ_c, the mean cluster radius r_c, and the number of clusters. Each histogram is shown twice, on the left hand side for the calculation of $\tilde{d}_c$ with weighted gamma distributions and on the right hand side or the calculation of $\tilde{d}_c$ with unweighted gamma distributions.

A.4.4 Results of Simulation Study in Detail

method	parameter	pattern 1	pattern 2	pattern 3	pattern 4
	p_c	0.4	0.4	0.4	0.4
	p_{mc}	0	0	0.6	0.6
	μ_c	4	8	4	8
	r_c	15	30	15	30
	$e(p_c)$	0.057	0.070	0.304	0.311
GAMMICS	$e(\mu_c)$	0.137	3.131	2.450	3.112
(weighted)	$e(r_c)$	8.707	6.418	16.745	10.228
	$e(p_c)$	0.057	0.070	0.304	0.311
GAMMICS	$e(\mu_c)$	0.218	3.020	0.908	2.666
(unweighted)	$e(r_c)$	7.730	5.574	9.693	9.174
	$e(p_c)$	0.105	0.058	0.067	0.129
DBSCAN	$e(\mu_c)$	1.399	5.157	1.042	4.280
($\varepsilon = 20$)	$e(r_c)$	3.238	9.649	5.764	8.518
	$e(p_c)$	0.556	0.545	0.552	0.593
DBSCAN	$e(\mu_c)$	17.851	7.734	14.080	9.850
($\varepsilon = 100$)	$e(r_c)$	253.792	198.907	160.869	139.526
	$e(p_c)$	0.313	0.247	0.429	0.433
DPM model	$e(\mu_c)$	0.901	5.066	2.005	0.340
(informative)	$e(r_c)$	6.083	20.546	21.534	9.776
	$e(p_c)$	0.522	0.515	0.548	0.596
DPM model	$e(\mu_c)$	4.099	0.934	3.005	6.160
(non-inform.)	$e(r_c)$	39.914	19.093	51.712	49.242
	$e(p_c)$	0.525	0.520	0.548	0.591
bDBSCAN	$e(\mu_c)$	0.099	4.066	1.005	2.340
	$e(r_c)$	46.932	35.157	48.234	38.453
H-function	$e(r_c)$	53.703	34.984	156.905	118.902
GAMMICS	$ar(\tilde{\alpha}_0)$	0.406	0.404	0.388	0.334
acceptance rate	$ar(\tilde{\alpha}_1)$	0.320	0.260	0.280	0.268

Table A.1: Estimates of the proportion of clustered points ($\hat{p}_c$), the mean cluster size ($\hat{\mu}_c$) and the mean cluster radius ($\hat{r}_c$), for four selected simulated point patterns with $p_c = 0.4$ differing in underlying values for p_{mc}, μ_c and r_c, for GAMMICS (with and without weighting of gamma distribution when calculating $\tilde{d}_c$), DBSCAN with $\varepsilon = 20, 100$ nm, the DPM model with informative and non-informative prior distributions, Bayesian DBSCAN and the H-function. The acceptance rates $ar(\tilde{\alpha}_0)$ and $ar(\tilde{\alpha}_1)$ for the shape parameters of the gamma distributions in GAMMICS can also be seen.

method	parameter	pattern 1	pattern 2	pattern 3	pattern 4
	p_c	0.8	0.8	0.8	0.8
	p_{mc}	0	0	0.6	0.6
	μ_c	4	8	4	8
	r_c	15	30	15	30
	$e(p_c)$	0.076	0.004	0.044	0.012
GAMMICS	$e(\mu_c)$	0.142	0.685	1.532	0.976
(weighted)	$e(r_c)$	9.986	3.297	10.591	5.024
	$e(p_c)$	0.076	0.004	0.044	0.012
GAMMICS	$e(\mu_c)$	0.318	2.670	0.085	3.423
(unweighted)	$e(r_c)$	6.214	5.965	2.389	12.272
	$e(p_c)$	0.369	0.320	0.217	0.105
DBSCAN	$e(\mu_c)$	1.358	4.710	0.142	3.123
($\varepsilon = 20$)	$e(r_c)$	2.788	9.857	9.613	1.180
	$e(p_c)$	0.186	0.163	0.167	0.172
DBSCAN	$e(\mu_c)$	13.613	8.855	14.488	16.456
($\varepsilon = 100$)	$e(r_c)$	205.032	161.970	139.942	128.833
	$e(p_c)$	0.487	0.032	0.104	0.126
DPM model	$e(\mu_c)$	0.874	0.534	1.059	1.989
(informative)	$e(r_c)$	3.506	9.269	23.026	18.399
	$e(p_c)$	0.129	0.153	0.153	0.197
DPM model	$e(\mu_c)$	2.126	1.466	2.059	3.489
(non-inform.)	$e(r_c)$	24.980	23.833	36.886	34.186
	$e(p_c)$	0.176	0.168	0.165	0.178
bDBSCAN	$e(\mu_c)$	1.126	0.466	2.059	0.989
	$e(r_c)$	44.811	37.626	49.587	39.404
H-function	$e(r_c)$	47.327	46.157	114.507	124.770
GAMMICS	$ar(\tilde{\alpha}_0)$	0.478	0.532	0.386	0.298
acceptance rate	$ar(\tilde{\alpha}_1)$	0.316	0.276	0.226	0.238

Table A.2: Median absolute estimation errors for four selected simulated point patterns with $p_c = 0.8$ differing in underlying values for p_{mc}, μ_c and r_c, for GAMMICS (with and without weighting of gamma distribution when calculating $\tilde{d}_c$), DBSCAN with $\varepsilon = 20, 100$ nm, the DPM model with informative and non-informative prior distributions, Bayesian DBSCAN and the H-function. In particular, $e(p_c)$ = median absolute estimation error for p_c, $e(\mu_c)$= median absolute estimation error for μ_c, $e(r_c)$ = median absolute estimation error for r_c. The acceptance rates $ar(\tilde{\alpha}_0)$ and $ar(\tilde{\alpha}_1)$ for the shape parameters of the gamma distributions in GAMMICS can also be seen.

method	μ_{mc}	r_{mc}	Λ	σ_d	mcr	$e(p_c)$	$e(\mu_c)$	$e(r_c)$	$e(r_c)_H$
DBSCAN ($\varepsilon = 20$ nm)	20	100 nm	125	20 nm	0.352	0.128	1.661	8.304	133.894
DBSCAN ($\varepsilon = 100$ nm)	20	100 nm	125	20 nm	0.467	0.466	21.020	197.247	133.894
DBSCAN ($\varepsilon = 20$ nm)	30	150 nm	125	20 nm	0.337	0.116	1.723	8.019	200.335
DBSCAN ($\varepsilon = 100$ nm)	30	150 nm	125	20 nm	0.474	0.439	24.551	217.173	200.335
DBSCAN ($\varepsilon = 20$ nm)	20	100 nm	50	20 nm	0.336	0.131	1.693	8.132	139.282
DBSCAN ($\varepsilon = 100$ nm)	20	100 nm	50	20 nm	0.419	0.409	11.233	123.971	139.282
DBSCAN ($\varepsilon = 20$ nm)	20	100 nm	200	20 nm	0.360	0.133	1.540	7.938	131.930
DBSCAN ($\varepsilon = 100$ nm)	20	100 nm	200	20 nm	0.493	0.490	55.265	269.819	131.930
DBSCAN ($\varepsilon = 20$ nm)	20	100 nm	275	20 nm	0.372	0.148	1.463	7.842	133.034
DBSCAN ($\varepsilon = 100$ nm)	20	100 nm	275	20 nm	0.496	0.426	125.425	339.644	133.034
DBSCAN ($\varepsilon = 20$ nm)	20	100 nm	125	0 nm	0.272	0.134	1.203	7.974	129.633
DBSCAN ($\varepsilon = 100$ nm)	20	100 nm	125	0 nm	0.464	0.459	17.886	167.197	129.633
DBSCAN ($\varepsilon = 20$ nm)	30	150 nm	125	0 nm	0.240	0.117	1.240	8.055	197.467
DBSCAN ($\varepsilon = 100$ nm)	30	150 nm	125	0 nm	0.474	0.474	21.765	189.441	197.467

Table A.3: Results of the simulation study for DBSCAN with $\varepsilon = 20, 100$ nm and the H-function (last column), where $E(\Lambda) = E(\text{points}/\mu m^2)$, i.e. the expected point density, σ_d = detection error, μ_{mc} = expected size of metaclusters, r_{mc} = expected radius of metaclusters, mcr = median misclassification rate w.r.t clustered vs. non-clustered points, $e(p_c)$ = median absolute estimation error for p_c, $e(\mu_c)$= median absolute estimation error for μ_c, $e(r_c)$ = median absolute estimation error for r_c, $e(r_c)_H$= median absolute estimation error for r_c obtained by H-function approach.

method	μ_{mc}	r_{mc}	Λ	σ_d	mcr	$e(p_c)$	$e(\mu_c)$	$e(r_c)$	$e(r_c)_H$
GAMMICS (weighted gamma distributions)	20	100 nm	125	20 nm	0.325	0.160	1.474	7.883	133.894
GAMMICS (unweighted gamma distributions)	20	100 nm	125	20 nm	0.325	0.160	1.477	6.576	133.894
GAMMICS (weighted gamma distributions)	30	150 nm	125	20 nm	0.308	0.163	1.489	8.299	200.335
GAMMICS (unweighted gamma distributions)	30	150 nm	125	20 nm	0.308	0.163	1.414	6.835	200.335
GAMMICS (weighted gamma distributions)	20	100 nm	50	20 nm	0.289	0.157	2.385	14.591	139.282
GAMMICS (unweighted gamma distributions)	20	100 nm	50	20 nm	0.289	0.157	1.642	9.297	139.282
GAMMICS (weighted gamma distributions)	20	100 nm	200	20 nm	0.340	0.180	1.313	7.165	131.930
GAMMICS (unweighted gamma distributions)	20	100 nm	200	20 nm	0.340	0.180	1.335	7.087	131.930
GAMMICS (weighted gamma distributions)	20	100 nm	275	20 nm	0.368	0.193	1.409	7.154	133.034
GAMMICS (unweighted gamma distributions)	20	100 nm	275	20 nm	0.368	0.193	1.515	6.985	133.034
GAMMICS (weighted gamma distributions)	20	100 nm	125	0 nm	0.244	0.113	1.529	6.254	129.633
GAMMICS (unweighted gamma distributions)	20	100 nm	125	0 nm	0.244	0.113	1.235	7.072	129.633
GAMMICS (weighted gamma distributions)	30	150 nm	125	0 nm	0.214	0.080	1.416	5.831	197.467
GAMMICS (unweighted gamma distributions)	30	150 nm	125	0 nm	0.214	0.080	1.173	6.870	197.467

Table A.4: Results of the simulation study for GAMMICS (with and without weighting of gamma distributions when calculating $\tilde{d}_c$), and the H-function (last column), where $E(\Lambda) = E(\text{points}/\mu m^2)$, i.e. the expected point density, σ_d = detection error, μ_{mc} = expected size of metaclusters, r_{mc} = expected radius of metaclusters, mcr = median misclassification rate w.r.t clustered vs. non-clustered points, $e(p_c)$ = median absolute estimation error for p_c, $e(\mu_c)$= median absolute estimation error for μ_c, $e(r_c)$ = median absolute estimation error for r_c, $e(r_c)_H$= median absolute estimation error for r_c obtained by H-function approach.

method	μ_{mc}	r_{mc}	Λ	σ_d	mcr	$e(p_c)$	$e(\mu_c)$	$e(r_c)$	$e(r_c)_H$
DPM model (informative priors)	20	100 nm	125	20 nm	0.363	0.339	3.715	25.063	133.894
DPM model (non-informative prior)	20	100 nm	125	20 nm	0.483	0.450	6.953	59.692	133.894
DPM model (informative prior)	30	150 nm	125	20 nm	0.361	0.291	3.019	21.826	200.335
DPM model (non-informative prior)	30	150 nm	125	20 nm	0.480	0.452	6.090	60.612	200.335
DPM model (informative prior)	20	100 nm	50	20 nm	0.333	0.334	3.166	30.676	139.282
DPM model (non-informative prior)	20	100 nm	50	20 nm	0.493	0.481	6.500	69.415	139.282
DPM model (informative prior)	20	100 nm	200	20 nm	0.377	0.338	3.248	22.514	131.930
DPM model (non-informative prior)	20	100 nm	200	20 nm	0.479	0.452	7.017	52.764	131.930
DPM model (informative prior)	20	100 nm	275	20 nm	0.388	0.313	3.155	21.675	133.034
DPM model (non-informative prior)	20	100 nm	275	20 nm	0.469	0.414	8.018	50.256	133.034
DPM model (informative prior)	20	100 nm	125	0 nm	0.327	0.228	2.026	8.362	129.633
DPM model (non-informative prior)	20	100 nm	125	0 nm	0.482	0.466	3.895	23.055	129.633
DPM model (informative prior)	30	150 nm	125	0 nm	0.308	0.181	1.839	6.715	197.467
DPM model (non-informative prior)	30	150 nm	125	0 nm	0.482	0.459	3.082	18.532	197.467

Table A.5: Results of the simulation study for the DPM model with informative and non-informative prior distributions and the H-function (last column), where $E(\Lambda) = E(\text{points}/\mu m^2)$, i.e. the expected point density, σ_d = detection error, μ_{mc} = expected size of metaclusters, r_{mc} = expected radius of metaclusters, mcr = median misclassification rate w.r.t clustered vs. non-clustered points, $e(p_c)$ = median absolute estimation error for p_c, $e(\mu_c)$= median absolute estimation error for μ_c, $e(r_c)$ = median absolute estimation error for r_c, $e(r_c)_H$= median absolute estimation error for r_c obtained by H-function approach.

Estimator	Error	
	Weighting	
	yes	no
$\hat{\mu}_c$ (mean)	1.539	1.395
$\hat{\mu}_c$ (median)	1.800	1.972
$\hat{r}_c$ (B) (mean)	7.395	7.120
$\hat{r}_c$ (B) (median)	14.172	16.023

Table A.6: Median absolute estimation errors for GAMMICS employing different estimators for μ_c and r_c. The estimates leading to the smallest median absolute estimation error are printed in red. For GAMMICS, the calculation of the radius for one cluster as one half of the maximum distance between any two points beloning to a given cluster was not considered in the new simulation study, since preliminary results showed that much better results could be obtained by the calculation of the radius of one cluster as the mean distance between any two points that belong to that cluster, multiplied with the factor $2 \cdot \frac{45\pi}{128}$. The shown estimators are described in more detail in Table A.10.

Estimator	Error	
	Prior	
	informative	non-informative
$\hat{\mu}_c$ (mean)	4.134	14.057
$\hat{\mu}_c$ (median)	2.879	6.000
$\hat{r}_c$ (A) (mean)	20.784	76.501
$\hat{r}_c$ (A) (median)	21.144	47.217
$\hat{r}_c$ (B) (mean)	56.132	143.793
$\hat{r}_c$ (B) (median)	58.966	106.181

Table A.7: Median absolute estimation errors for the DPM model employing different estimators for μ_c and r_c. The estimates leading to the smallest median absolute estimation error are printed in red. The shown estimators are described in more detail in Table A.10.

Estimator	Error
$\hat{p}_c$	0.349
$\hat{\mu}_c$ (mean)	2.223
$\hat{\mu}_c$ (median)	1.066
$\hat{r}_c$ (A) (mean)	40.062
$\hat{r}_c$ (A) (median)	42.108
$\hat{r}_c$ (B) (mean)	98.997
$\hat{r}_c$ (B) (median)	102.296

Table A.8: Median absolute estimation errors for the Bayesian DBSCAN model employing different estimators for μ_c and r_c. The estimates for μ_c and r_c leading to the smallest median absolute estimation error are printed in red. The reported values for Bayesian DBSCAN are based on results for only eight simulated point patterns as reported in Tables A.1 and A.2, not on the whole simulation study. The estimation error for $\hat{p}_c$ was additionally reported across all considered point patterns. The misclassification rate is not obtained for Bayesian DBSCAN since Bayesian DBSCAN does not provide the points' cluster allocations for the MCMC iterations, needed for classification. The shown estimators are described in more detail in Table A.10.

Estimator	Error													
	ε value													
	5	10	15	20	25	30	35	40	45	50	55	60	...	100
mcr	0.42	0.38	0.34	0.33	0.32	0.32	0.33	0.35	0.36	0.37	0.39	0.40	...	0.43
$\hat{p}_c$	0.38	0.23	**0.13**	**0.13**	0.17	0.23	0.28	0.32	0.34	0.36	0.38	0.39	...	0.42
$\hat{\mu}_c$ (mean)	4.68	2.68	1.80	**1.51**	**1.40**	1.76	2.38	3.35	4.93	7.08	9.34	11.44	...	27.98
$\hat{\mu}_c$ (median)	4.98	3.42	2.17	2.12	2.07	1.94	**1.23**	**1.10**	**1.08**	**1.11**	**1.18**	1.80	...	2.98
$\hat{r}_c$ (A) (mean)	19.98	18.01	15.05	9.21	7.83	7.66	9.21	14.24	21.90	30.26	38.73	47.17	...	117.35
$\hat{r}_c$ (A) (median)	19.88	17.88	15.72	12.52	8.11	8.60	7.64	7.94	8.54	12.15	16.43	20.36	...	52.78
$\hat{r}_c$ (B) (mean)	14.18	13.62	9.85	8.03	11.25	20.91	32.68	45.52	59.64	73.78	88.01	102.65	...	215.03
$\hat{r}_c$ (B) (median)	14.01	13.32	11.89	8.69	8.07	13.72	20.58	28.23	36.71	46.27	54.25	62.22	...	118.31

Table A.9: Errors for DBSCAN employing different estimators for μ_c and r_c and different values for ε. mcr = misclassification rate. For $\hat{\mu}$ and $\hat{r}_c$, the median absolute estimation error is given. The estimates for μ_c and r_c leading to the smallest median absolute estimation error are printed in red. The errors printed in bold numbers correspond to cases in which GAMMICS is outperformed. The shown estimators are described in more detail in Table A.10.

Estimator	Description
$\hat{\mu}_c$ (mean)	μ_c is estimated (if applicable, in each MCMC iteration) as the mean size across all clusters.
$\hat{\mu}_c$ (median)	μ_c is estimated (if applicable, in each MCMC iteration) as the median size across all clusters.
$\hat{r}_c$ (A) (mean)	r_c is estimated (if applicable, in each MCMC iteration) as the mean radius across all clusters, where the radius for each cluster is calculated as one half of the maximum distance between any two points belonging to that cluster.
$\hat{r}_c$ (A) (median)	r_c is estimated (if applicable, in each MCMC iteration) as the median radius across all clusters, where the radius for each cluster is calculated as one half of the maximum distance between any two points belonging to that cluster.
$\hat{r}_c$ (B) (mean)	r_c is estimated (if applicable, in each MCMC iteration) as the mean radius across all clusters, where the radius for each cluster is calculated as the mean distance between any two points that belong to that cluster, multiplied with the factor $2 \cdot \frac{45\pi}{128}$.
$\hat{r}_c$ (B) (median)	r_c is estimated (if applicable, in each MCMC iteration) as the mean radius across all clusters, where the radius for each cluster is calculated as the median distance between any two points that belong to that cluster, multiplied with the factor $2 \cdot \frac{45\pi}{128}$.

Table A.10: Different estimators for μ_c and r_c considered in the simulation study.

method	param.	ROI 1	ROI 2	ROI 3	ROI 4	ROI 5	ROI 6
	Λ	51.369	56.719	76.262	94.113	116.768	146.137
	$\hat{p}_c$	0.750	0.767	0.716	0.664	0.650	0.601
GAMMICS	$\hat{\mu}_c$	5.214	5.273	5.592	5.125	5.902	4.497
(weighted)	$\hat{r}_c$	23.371	27.975	25.910	23.460	20.527	18.435
	$\hat{p}_c$	0.750	0.767	0.716	0.664	0.650	0.601
GAMMICS	$\hat{\mu}_c$	5.014	4.844	4.916	4.364	5.041	4.450
(unweighted)	$\hat{r}_c$	22.952	23.527	20.124	19.363	17.657	18.382
	$\hat{p}_c$	0.586	0.549	0.558	0.566	0.629	0.593
DBSCAN	$\hat{\mu}_c$	4.709	4.161	3.974	4.029	5.056	4.433
($\varepsilon = 20$)	$\hat{r}_c$	25.874	23.751	24.363	24.366	26.461	24.861
	$\hat{p}_c$	0.960	0.951	0.969	0.966	0.976	0.979
DBSCAN	$\hat{\mu}_c$	7.784	7.841	9.396	12.341	15.467	22.595
($\varepsilon = 100$)	$\hat{r}_c$	86.915	94.654	113.696	137.107	140.046	188.619
	$\hat{p}_c$	0.840	0.821	0.837	0.825	0.881	0.878
DPM model	$\hat{\mu}_c$	5.000	5.000	5.000	5.000	6.000	6.000
(informative)	$\hat{r}_c$	29.115	28.167	29.489	30.045	35.106	38.577
	$\hat{p}_c$	0.921	0.910	0.957	0.959	0.971	0.955
DPM model	$\hat{\mu}_c$	6.000	5.000	8.000	8.000	8.000	10.000
(non-inform.)	$\hat{r}_c$	38.587	36.391	42.666	46.412	45.925	61.799
	$\hat{p}_c$	0.970	0.972	0.969	0.971	0.981	0.980
bDBSCAN	$\hat{\mu}_c$	5.000	5.000	6.000	6.000	6.000	8.000
	$\hat{r}_c$	48.725	54.059	56.291	58.826	59.318	75.199
H-function	$\hat{r}_c$	71.055	62.695	60.605	68.965	71.055	75.234
GAMMICS	$ar(\tilde{\alpha}_0)$	0.432	0.514	0.370	0.326	0.274	0.270
accept. rate	$ar(\tilde{\alpha}_1)$	0.284	0.268	0.278	0.250	0.230	0.220

Table A.11: Estimates of the proportion of clustered points ($\hat{p}_c$), the mean cluster size ($\hat{\mu}_c$) and the mean cluster radius ($\hat{r}_c$) for six selected regions of interest from the experimental data differing in their point density Λ (in points/μm^2). Compared methods are GAMMICS (with and without weighting of gamma distribution when calculating $\tilde{d}_c$), DBSCAN with $\varepsilon = 20, 100$ nm, the DPM model with informative and non-informative prior distributions, Bayesian DBSCAN and the H-function. The acceptance rates $ar(\alpha_1)$ and $ar(\alpha_2)$ for the shape parameters of the gamma distributions in GAMMICS are also given.

A.5 BUGS Model: Full Conditional Distributions

Please also refer to Section 5.3.3 for notation and definitions used in the following.

With $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_K)'$, $\boldsymbol{\sigma}^2 = (\sigma_1^2, \ldots, \sigma_K^2)'$ and $\boldsymbol{\pi}^* = (\pi_1^*, \ldots, \pi_K^*)'$, the joint likelihood is

$$p(\boldsymbol{Z}, \boldsymbol{T}, \boldsymbol{\mu}, \boldsymbol{\sigma}^2, \boldsymbol{\pi}^*, m_{\mu_1}, \ldots, m_{\mu_{K/2-1}}, m_{\mu_{K/2+2}}, \ldots, m_{\mu_K}, v_{\mu_1}^2, \ldots, v_{\mu_{K/2-1}}^2,$$

$$v_{\mu_{K/2+2}}^2, \ldots, v_{\mu_K}^2, a_{\sigma_1}, \ldots, a_{\sigma_K}, b_{\sigma_1}, \ldots, b_{\sigma_K}, \boldsymbol{\alpha}) \propto$$

$$p(\boldsymbol{Z}|\boldsymbol{\pi}^*, \boldsymbol{\mu}, \boldsymbol{\sigma}^2) \cdot \prod_{k=1,\ldots,K/2-1,K/2+2,\ldots,K} p(\mu_k|m_{\mu_k}, v_{\mu_k}^2) \cdot \prod_{k=1}^{K} p(\sigma_k^2|a_{\sigma_k}, b_{\sigma_k}) \cdot p(\boldsymbol{T}|\boldsymbol{\pi}^*) \cdot p(\boldsymbol{\pi}^*|\boldsymbol{\alpha})$$

where

$$\begin{aligned}
p(\boldsymbol{T}|\boldsymbol{\pi}^*) &\propto \prod_{i=1}^{n} \left(\prod_{k=1}^{K} {\pi_i^*}^{\mathbf{I}(T_i=k)} \right), \\
p(\boldsymbol{Z}|\boldsymbol{\pi}^*, \boldsymbol{\mu}, \boldsymbol{\sigma}^2) &\propto \prod_{i=1}^{n} p(Z_i|\boldsymbol{\pi}^*, \boldsymbol{\mu}, \boldsymbol{\sigma}^2) \\
&= \prod_{i=1}^{n} \left(\sum_{k=1}^{K/2-1} \pi_k^* \cdot p(Z_i|\mu_k, \sigma_k^2) + \sum_{k=K/2}^{K/2+1} \pi_k^* \cdot p(Z_i|\sigma_k^2) + \right. \\
&\qquad \left. \sum_{k=K/2+2}^{K} \pi_k^* \cdot p(Z_i|\mu_k, \sigma_k^2) \right),
\end{aligned}$$

and

$$
\begin{aligned}
p(Z_i|\mu_k,\sigma_k^2) &= \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{(Z_i-\mu_k)^2}{2\sigma_k^2}}, \quad k=1,\dots K/2-1, K/2+2,\dots K,\\
p(Z_i|\sigma_k^2) &= \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{Z_i^2}{2\sigma_k^2}}, \quad k=K/2, K/2+1,\\
p(\mu_k|m_{\mu_k},v_{\mu_k}^2) &\propto \frac{1}{\sqrt{2\pi v_{\mu_k}^2}} e^{-\frac{(\mu_k-m_{\mu_k})^2}{2v_{\mu_k}^2}} \cdot \mathbf{I}(\mu_k<0) + \delta_{<0} \cdot \mathbf{I}(\mu_k=0),\\
&\qquad\qquad k=1,\dots,K/2-1,\\
p(\mu_k|m_{\mu_k},v_{\mu_k}^2) &\propto \frac{1}{\sqrt{2\pi v_{\mu_k}^2}} e^{-\frac{(\mu_k-m_{\mu_k})^2}{2v_{\mu_k}^2}} \cdot \mathbf{I}(\mu_k>0) + \delta_{>0} \cdot \mathbf{I}(\mu_k=0),\\
&\qquad\qquad k=K/2+2,\dots,K,\\
p\left(\frac{1}{\sigma_k^2}\middle| a_{\sigma_k}, b_{\sigma_k}\right) &\propto \frac{1}{b_{\sigma_k}^{a_{\sigma_k}}\Gamma(a_{\sigma_k})}\left(\frac{1}{\sigma_k^2}\right)^{a_{\sigma_k}-1} e^{-\frac{1}{\sigma_k^2 b_{\sigma_k}}}, \quad k=1,\dots,K,\\
p(\boldsymbol{\pi}^*|\boldsymbol{\alpha}) &\propto \pi_1^{*\alpha_1-1}\cdots\pi_K^{*\alpha_K-1}\cdot\frac{\Gamma(\sum_{k=1}^K\alpha_k)}{\prod_{k=1}^K\Gamma(\alpha_k)},
\end{aligned}
$$

where

$$
\begin{aligned}
\delta_{<0} &= 1-\int_{-\infty}^{0}\frac{1}{\sqrt{2\pi v_{\mu_k}^2}} e^{-\frac{(\mu_k-m_{\mu_k})^2}{2v_{\mu_k}^2}} d\mu_k,\\
\delta_{>0} &= 1-\int_{0}^{\infty}\frac{1}{\sqrt{2\pi v_{\mu_k}^2}} e^{-\frac{(\mu_k-m_{\mu_k})^2}{2v_{\mu_k}^2}} d\mu_k,
\end{aligned}
$$

and $\mathbf{I}()$ is the indicator function. The model is fitted with the WinBUGS software (Spiegelhalter *et al.*, 2003).

A.6 BUGS model: Additional Results

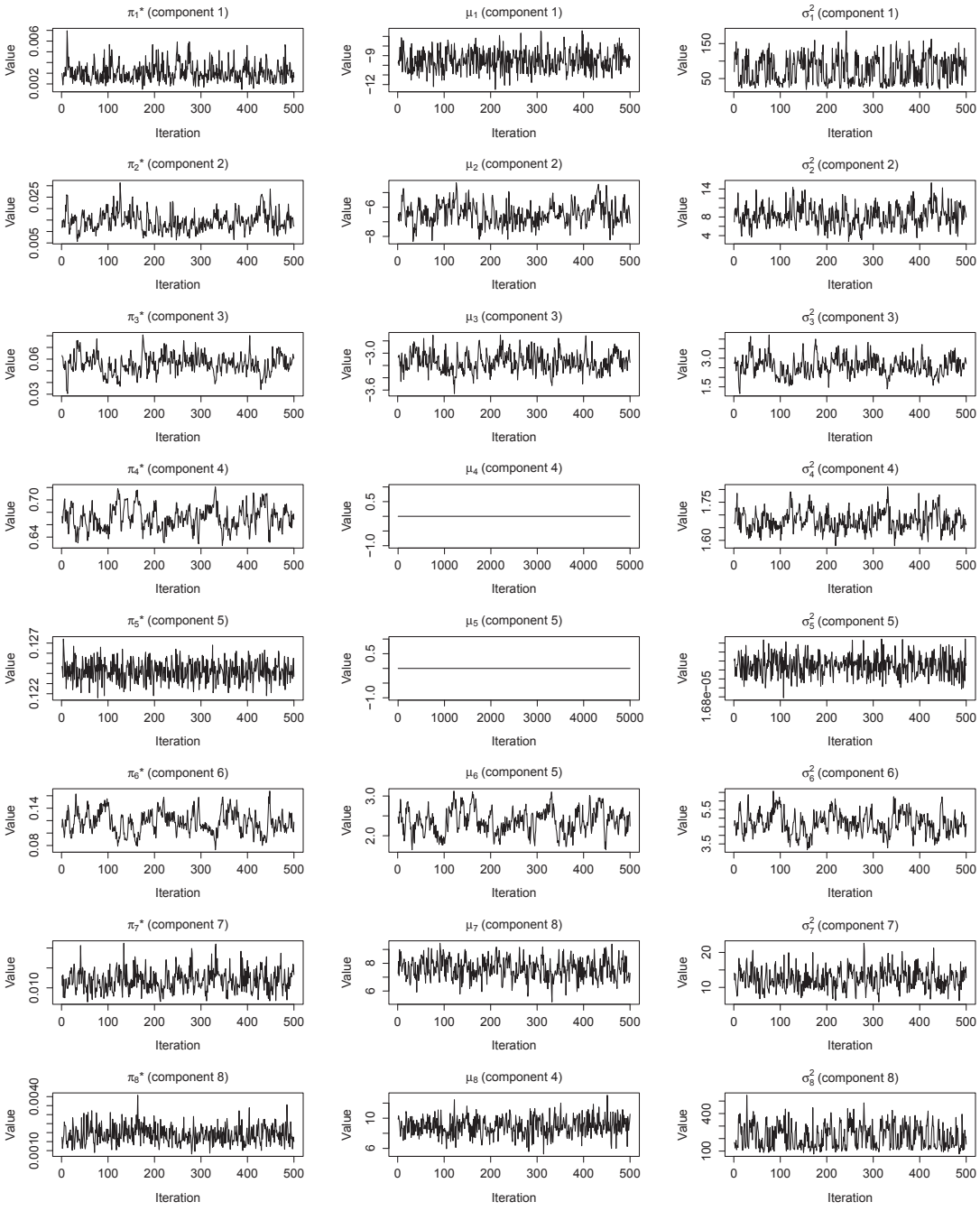

Figure A.15: Trace plots of the model parameters after fitting the Bayesian mixture model employing normal components (see Section 5.3.3) to the BCR-ABL data set. The first 2 000 iterations were discarded and the remaining iterations were thinned by keeping every 150th iteration. The remaining 500 iterations were used for parameter estimation and are shown in the trace plots.

A.7 Epigenomix Model: Model Details

Please also refer to Section 5.4.3 for notation and definitions used in the following.

A.7.1 Full Conditional Distributions for Model (5.6)

With $\boldsymbol{\lambda} = (\lambda_1, \lambda_K)'$, $\boldsymbol{\sigma}^2 = (\sigma_2^2, \ldots, \sigma_{K-1}^2)'$ and $\boldsymbol{\pi^*} = (\pi_1^*, \ldots, \pi_K^*)'$, the joint likelihood is

$$p(\boldsymbol{Z}, \boldsymbol{T}, \boldsymbol{\lambda}, \boldsymbol{\sigma}^2, \boldsymbol{\pi^*}, a_{\lambda_1}, b_{\lambda_1}, a_{\lambda_K}, b_{\lambda_K}, a_{\sigma_2}, \ldots, a_{\sigma_{K-1}}, b_{\sigma_2}, \ldots, b_{\sigma_{K-1}}, \alpha^*, a_{\alpha^*}, b_{\alpha^*}) \propto$$

$$p(\boldsymbol{Z}|\boldsymbol{\pi^*}, \boldsymbol{\lambda}, \boldsymbol{\sigma}^2) \cdot \prod_{k=1,K} p(\lambda_k|a_{\lambda_k}, b_{\lambda_k}) \cdot \prod_{k=2}^{K-1} p(\sigma_k^2|a_{\sigma_k}, b_{\sigma_k}) \cdot p(\boldsymbol{T}|\boldsymbol{\pi^*}) \cdot p(\boldsymbol{\pi^*}|\alpha^*) \cdot p(\alpha^*|a_{\alpha^*}, b_{\alpha^*}),$$

where

$$\begin{aligned}
p(\boldsymbol{T}|\boldsymbol{\pi^*}) &\propto \prod_{i=1}^{n} \left(\prod_{k=1}^{K} {\pi_i^*}^{\mathbf{I}(T_i=k)} \right), \\
p(\boldsymbol{Z}|\boldsymbol{\pi^*}, \boldsymbol{\lambda}, \boldsymbol{\sigma}^2) &\propto \prod_{i=1}^{n} p(Z_i|\boldsymbol{\pi^*}, \boldsymbol{\lambda}, \boldsymbol{\sigma}^2) \\
&= \prod_{i=1}^{n} \left(\pi_1^* \cdot p(-Z_i|\lambda_1) + \sum_{k=2}^{K-1} \pi_k^* \cdot p(Z_i|\sigma_k^2) + \pi_K^* \cdot p(Z_i|\lambda_K) \right) \\
&= \prod_{i=1}^{n} \left(\pi_1^* \cdot \lambda_1 e^{\lambda_1 Z_i} + \sum_{k=2}^{K-1} \pi_k^* \cdot \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{Z_i^2}{2\sigma_k^2}} + \pi_K^* \cdot \lambda_K e^{-\lambda_K Z_i} \right)
\end{aligned}$$

and

$$\begin{aligned}
p(\lambda_k|a_{\lambda_k}, b_{\lambda_k}) &\propto \frac{1}{b_{\lambda_k}^{a_{\lambda_k}} \Gamma(a_{\lambda_k})} \lambda_k^{a_{\lambda_k}-1} e^{-\frac{\lambda_k}{b_{\lambda_k}}}, \quad k = 1, K, \\
p\left(\frac{1}{\sigma_k^2} \middle| a_{\sigma_k}, b_{\sigma_k}\right) &\propto \frac{1}{b_{\sigma_k}^{a_{\sigma_k}} \Gamma(a_{\sigma_k})} \left(\frac{1}{\sigma_k^2}\right)^{a_{\sigma_k}-1} e^{-\frac{1}{\sigma_k^2 b_{\sigma_k}}}, \quad k = 2, \ldots, K-1, \\
p(\boldsymbol{\pi^*}|\alpha^*) &\propto {\pi_1^*}^{\alpha^*/K-1} \cdots {\pi_K^*}^{\alpha^*/K-1} \cdot \frac{\Gamma(\sum_{k=1}^{K} \frac{\alpha^*}{K})}{\prod_{k=1}^{K} \Gamma\left(\frac{\alpha^*}{K}\right)}, \\
p(\alpha^*|a_{\alpha^*}, b_{\alpha^*}) &\propto \frac{1}{b_{\alpha^*}^{a_{\alpha^*}} \Gamma(a_{\alpha^*})} (\alpha^*)^{a_{\alpha^*}-1} e^{-\frac{\alpha^*}{b_{\alpha^*}}},
\end{aligned}$$

where $\mathbf{I}()$ is the indicator function.

A.7.2 Full Conditional Distributions for Model (5.7)

With $\tilde{\boldsymbol{\alpha}} = (\tilde{\alpha}_1, \tilde{\alpha}_K)'$, $\tilde{\boldsymbol{\beta}} = (\tilde{\beta}_1, \tilde{\beta}_K)'$, $\boldsymbol{\sigma}^2 = (\sigma_2^2, \ldots, \sigma_{K-1}^2)'$ and $\boldsymbol{\pi^*} = (\pi_1^*, \ldots, \pi_K^*)'$, the joint likelihood is

$$p(\boldsymbol{Z}, \boldsymbol{T}, \tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\beta}}, \boldsymbol{\sigma}^2, \boldsymbol{\pi^*}, a_{\tilde{\alpha}_1}, a_{\tilde{\alpha}_K}, b_{\tilde{\alpha}_1}, b_{\tilde{\alpha}_K}, c_{\tilde{\beta}_1}, c_{\tilde{\beta}_K}, d_{\tilde{\beta}_1}, d_{\tilde{\beta}_K}, a_{\sigma_2}, \ldots, a_{\sigma_{K-1}}, b_{\sigma_2}, \ldots, b_{\sigma_{K-1}}, \alpha^*, a_{\alpha^*}, b_{\alpha^*}) \propto$$

$$p(\boldsymbol{Z}|\boldsymbol{\pi^*}, \tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\beta}}, \boldsymbol{\sigma}^2) \cdot \prod_{k=1,K} p(\tilde{\alpha}_k|a_{\tilde{\alpha}_k}, b_{\tilde{\alpha}_k}) \cdot \prod_{k=1,K} p\left(\frac{1}{\tilde{\beta}_k}\middle| c_{\tilde{\beta}_k}, d_{\tilde{\beta}_k}\right) \cdot \prod_{k=2}^{K-1} p(\sigma_k^2|a_{\sigma_k}, b_{\sigma_k})$$
$$\cdot\, p(\boldsymbol{T}|\boldsymbol{\pi^*}) \cdot p(\boldsymbol{\pi^*}|\alpha^*) \cdot p(\alpha^*|a_{\alpha^*}, b_{\alpha^*}),$$

where

$$\begin{aligned}
p(\boldsymbol{T}|\boldsymbol{\pi^*}) &\propto \prod_{i=1}^{n}\left(\prod_{k=1}^{K} \pi_i^{*\mathbf{I}(T_i=k)}\right), \\
p(\boldsymbol{Z}|\boldsymbol{\pi^*}, \tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\beta}}, \boldsymbol{\sigma}^2) &\propto \prod_{i=1}^{n} p(Z_i|\boldsymbol{\pi^*}, \tilde{\boldsymbol{\alpha}}, \tilde{\boldsymbol{\beta}}, \boldsymbol{\sigma}^2) \\
&= \prod_{i=1}^{n}\left(\pi_1^* \cdot p(-Z_i|\tilde{\alpha}_1, \tilde{\beta}_1) + \sum_{k=2}^{K-1} \pi_k^* \cdot p(Z_i|\sigma_k^2) + \pi_K^* \cdot p(Z_i|\tilde{\alpha}_K, \tilde{\beta}_K)\right) \\
&= \prod_{i=1}^{n}\left(\pi_1^* \cdot \frac{1}{\Gamma(\tilde{\alpha}_1)\tilde{\beta}_1^{\tilde{\alpha}_1}}(-1) \cdot Z_i^{\tilde{\alpha}_1 - 1} e^{\frac{Z_i}{\tilde{\beta}_1}} + \sum_{k=2}^{K-1} \pi_k^* \cdot \frac{1}{\sqrt{2\pi\sigma_k^2}} e^{-\frac{Z_i^2}{2\sigma_k^2}} + \right. \\
&\qquad \left. \pi_K^* \cdot \frac{1}{\Gamma(\tilde{\alpha}_K)\tilde{\beta}_K^{\alpha_2}} Z_i^{\tilde{\alpha}_K - 1} e^{-\frac{Z_i}{\tilde{\beta}_K}}\right)
\end{aligned}$$

and

$$\begin{aligned}
p(\tilde{\alpha}_k|a_{\tilde{\alpha}_k}, b_{\tilde{\alpha}_k}) &\propto \frac{1}{\Gamma(a_{\tilde{\alpha}_k}) b_{\tilde{\alpha}_k}^{a_{\tilde{\alpha}_k}}} \tilde{\alpha}_k^{a_{\tilde{\alpha}_k} - 1} e^{-\frac{\tilde{\alpha}_k}{b_{\tilde{\alpha}_k}}}, \quad k = 1, K, \\
p\left(\frac{1}{\tilde{\beta}_k}\middle| c_{\tilde{\beta}_k}, d_{\tilde{\beta}_k}\right) &\propto \frac{1}{d_{\tilde{\beta}_k}^{c_{\tilde{\beta}_k}} \Gamma(c_{\tilde{\beta}_k})} \left(\frac{1}{\tilde{\beta}_k}\right)^{c_{\tilde{\beta}_k} - 1} e^{-\frac{1}{\tilde{\beta}_k \cdot d_{\tilde{\beta}_k}}}, \quad k = 1, K, \\
p\left(\frac{1}{\sigma_k^2}\middle| a_{\sigma_k}, b_{\sigma_k}\right) &\propto \frac{1}{b_{\sigma_k}^{a_{\sigma_k}} \Gamma(a_{\sigma_k})} \left(\frac{1}{\sigma_k^2}\right)^{a_{\sigma_k} - 1} e^{-\frac{1}{\sigma_k^2 b_{\sigma_k}}}, \quad k = 2, \ldots, K-1, \\
p(\boldsymbol{\pi^*}|\alpha^*) &\propto {\pi_1^*}^{\alpha^*/K - 1} \cdots {\pi_K^*}^{\alpha^*/K - 1} \cdot \frac{\Gamma(\sum_{k=1}^{K} \frac{\alpha^*}{K})}{\prod_{k=1}^{K} \Gamma\left(\frac{\alpha^*}{K}\right)}, \\
p(\alpha^*|a_{\alpha^*}, b_{\alpha^*}) &\propto \frac{1}{b_{\alpha^*}^{a_{\alpha^*}} \Gamma(a_{\alpha^*})} (\alpha^*)^{a_{\alpha^*} - 1} e^{-\frac{\alpha^*}{b_{\alpha^*}}},
\end{aligned}$$

where $\mathbf{I}()$ is the indicator function.

A.8 Epigenomix Model: Sampling Scheme

Please also refer to Section 5.4.3 for notation and definitions used in the following.

A.8.1 Sampling Scheme for Model (5.6)

The full sampling scheme is as follows:

- Resample the allocation (classification) indices $t_1, \ldots, t_n$ independently with probabilities

$$p(T_i = k|z_i, \boldsymbol{\pi}^*) \propto \begin{cases} \pi_k^* \cdot \text{Exp}(-z_i|\lambda_1) & k = 1 \\ \pi_k^* \cdot N(z_i|\sigma_k^2) & k = 2, \ldots, K-1 \\ \pi_k^* \cdot \text{Exp}(z_i|\lambda_K) & k = K. \end{cases}$$

  This reconfigures the n points independently among the K components, for which counts $n_k = |\{t_i = k, i = 1, \ldots, n\}|$, $k = 1, \ldots, K$, result. Note that some components may be empty, i.e. it may be $n_k = 0$ for some k.

- For $k = 1$, update $\alpha_k = \alpha^*/K + n_k$. Then draw a new vector $\boldsymbol{\pi}^*$ from Dirichlet$(\alpha_1, \ldots, \alpha_K)$.

- For $k = 2, \ldots, K-1$, update $a_{\sigma_k} = a_{\sigma_k,0} + \frac{n_k}{2}$ and $b_{\sigma_k} = \frac{b_{\sigma_k,0}}{1+\frac{b_{\sigma_k,0}}{2}\cdot\sum_{i:t_i=k} z_i}$, where $a_{\sigma_k,0}$ and $b_{\sigma_k,0}$ are the prior choices for a_{σ_k} and b_{σ_k}. Then draw $1/\sigma_k^2$ from Gamma(a_{σ_k},b_{σ_k}). If $n_k = 0$, draw from Gamma($a_{\sigma_k,0}$,$b_{\sigma_k,0}$).

- For $k = 1, K$, update $a_{\lambda_k} = a_{\lambda_k,0} + n_k$ and $b_{\lambda_k} = \frac{b_{\lambda_k,0}}{1+b_{\lambda_k,0}\cdot\sum_{i:t_i=k} z_i}$, where $a_{\lambda_k,0}$ and $b_{\lambda_k,0}$ are the prior choices for a_{λ_k} and b_{λ_k}. Then draw λ_k from Gamma(a_{λ_k},b_{λ_k}). If $n_k = 0$, draw from Gamma($a_{\lambda_k,0}$,$b_{\lambda_k,0}$).

- Resample the Dirichlet mass parameter α^* from Gamma(a_{α^*},b_{α^*}) via a Metropolis-Hastings step, where the proposal distribution is a normal distribution centered at the current value and mirrored at 0 with a predefined standard deviation.

A.8.2 Sampling Scheme for Model (5.7)

The full sampling scheme is as follows:

- Resample the allocation (classification) indices $t_1, \ldots, t_n$ independently with probabilities

$$p(T_i = k | z_i, \boldsymbol{\pi}^*) \propto \begin{cases} \pi_k^* \cdot \text{Gamma}(-z_i | \tilde{\alpha}_1, \tilde{\beta}_1) & k = 1 \\ \pi_k^* \cdot N(z_i | \sigma_k^2) & k = 2, \ldots, K-1 \\ \pi_k^* \cdot \text{Gamma}(z_i | \tilde{\alpha}_K, \tilde{\beta}_K) & k = K. \end{cases}$$

  This reconfigures the n points independently among the K components, for which counts $n_k = |\{t_i = k, i = 1, \ldots, n\}|$, $k = 1, \ldots, K$, result. Note that some components may be empty, i.e. it may be $n_k = 0$ for some k.

- For $k = 1$, update $\alpha_k = \alpha^*/K + n_k$. Then draw a new vector $\boldsymbol{\pi}^*$ from Dirichlet$(\alpha_1, \ldots, \alpha_K)$.

- For $k = 2, \ldots, K-1$, update $a_{\sigma_k} = a_{\sigma_k,0} + \frac{n_k}{2}$ and $b_{\sigma_k} = \frac{b_{\sigma_k,0}}{1 + \frac{b_{\sigma_k,0}}{2} \cdot \sum_{i:t_i=k} z_i}$, where $a_{\sigma_k,0}$ and $b_{\sigma_k,0}$ are the prior choices for a_{σ_k} and b_{σ_k}. Then draw $1/\sigma_k^2$ from Gamma(a_{σ_k},b_{σ_k}). If $n_k = 0$, draw from Gamma($a_{\sigma_k,0}$,$b_{\sigma_k,0}$).

- For $k = 1, K$, update $c_{\tilde{\beta}_k} = \tilde{\alpha}_k \cdot n_k + c_{\tilde{\beta}_k,0}$ and $d_{\tilde{\beta}_k} = \frac{d_{\tilde{\beta}_k 0}}{1 + d_{\tilde{\beta}_k 0} \cdot \sum_{i:t_i=k} z_i}$, where $c_{\tilde{\beta}_k,0}$ and $d_{\tilde{\beta}_k,0}$ are the prior choices for $c_{\tilde{\beta}_k}$ and $d_{\tilde{\beta}_k}$. Then draw $1/\tilde{\beta}_k$ from Gamma($c_{\tilde{\beta}_k}$,$d_{\tilde{\beta}_k}$). If $n_k = 0$, draw from Gamma($c_{\tilde{\beta}_k,0}$,$d_{\tilde{\beta}_k,0}$).

- For $k = 1, K$, draw $\tilde{\alpha}_k$ from Gamma($a_{\tilde{\alpha}_k,0}$,$b_{\tilde{\alpha}_k,0}$) in a Metropolis-Hastings step, where the proposal distribution is a normal distribution centered at the current value and mirrored at 0 with a predefined standard deviation.

- Resample the Dirichlet mass parameter α^* from Gamma(a_{α^*},b_{α^*}) via a Metropolis-Hastings step, where the proposal distribution is a normal distribution centered at the current value and mirrored at 0 with a predefined standard deviation.

A.9 Normalization Methods for ChIP-seq Data

This description is taken from the supplementary material of Klein *et al.* (2014).

Four different normalization methods were compared (see Figure A.16A). In the following text, $\mathsf{Y}_{ij},\ i = 1, \ldots, n,\ j = 1, \ldots, m$, denote the number of reads from sample j that are aligned within genomic region $\mathcal{R}_i$ (e.g. promoter regions). Duplicated reads as well as reads that aligned at multiple positions are removed prior to calculating Y_{ij}. $Y_{ij},\ i = 1, \ldots, n,\ j = 1, \ldots, m$, denote the normalized ChIP-seq values.

Scaling based on total counts

An obvious source of technical variation between ChIP-seq samples is a varying sequencing depth, i.e. the total number of sequenced DNA fragments. Many peak detection algorithms like MACS (Zhang *et al.*, 2008) or ChIPDiff (Xu *et al.*, 2008) scale each sample j by a factor s_j such that each sample has the same number of reads after normalization. Let N_j be the total number of unique reads obtained from sample j that could be uniquely aligned at an arbitrary position in the reference genome, then

$$s_j = \frac{1/m \sum_{l=1}^{m} N_l}{N_j}, \quad j = 1, \ldots, m,$$

and

$$Y_{ij} = \mathsf{Y}_{ij}\, s_j, \quad i = 1, \ldots, n,\ j = 1, \ldots, m.$$

Scaling based on region counts

Instead of using the total number of aligned reads, a scaling factor s_j can be calculated considering only those reads that were aligned within the regions of interest $\mathcal{R}_i,\ i = 1, \ldots, n$. Reads outside of any region $\mathcal{R}_i$ are more likely to be

background reads and are not considered in the analysis after normalization. Then

$$s_j = \frac{1/m \sum_{l=1}^{m} \sum_{i=1}^{n} \mathsf{Y}_{il}}{\sum_{i=1}^{n} \mathsf{Y}_{ij}}, \quad j = 1, \ldots, m,$$

and

$$Y_{ij} = \mathsf{Y}_{ij}\, s_j, \quad i = 1, \ldots, n,\ j = 1, \ldots, m.$$

Scaling based on median of ratios

Anders and Huber, 2010, proposed setting the scaling factor s_j to the median of the ratios of observed numbers of reads. Their estimation of the scaling factor is more robust in cases where some samples have more histone modifications within some of the regions $\mathcal{R}_i$ and consequently more reads in these regions than other samples. Originally, this approach was designed for normalizing RNA-seq data. In this case

$$s_j = \underset{i}{\text{median}}\ \frac{(\prod_{l=1}^{m} \mathsf{Y}_{il})^{1/m}}{\mathsf{Y}_{ij}}, \quad j = 1, \ldots, m,$$

and

$$Y_{ij} = \mathsf{Y}_{ij}\, s_j, \quad i = 1, \ldots, n,\ j = 1, \ldots, m.$$

Quantile normalization

Quantile normalization was successfully used to normalize microarray gene expression data (Bolstad *et al.*, 2003) and recently also applied to ChIP-seq data (Schenk *et al.*, 2012; Nair *et al.*, 2012). Here, quantile normalization is applied directly to the number of observed reads within regions Y_{ij}. Let $\tau_{\cdot j}$ be a permutation of indices for sample j such that $\mathsf{Y}_{(\tau_{1j})j} \leq \ldots \leq \mathsf{Y}_{(\tau_{nj})j}$. The quantile

normalized values are then defined as

$$Y_{(\tau_{ij})j} = 1/m \sum_{l=1}^{m} \mathsf{Y}_{(\tau_{il})l}, \quad i = 1, \ldots, n,\ j = 1, \ldots, m.$$

Quantile normalization assumes that the ChIP-seq values of all samples follow the same distribution. This assumption may not hold, e.g., if some samples were treated with an epigenetic drug causing modifications of histones genome wide.

A.10 Epigenomix Model: Additional Results

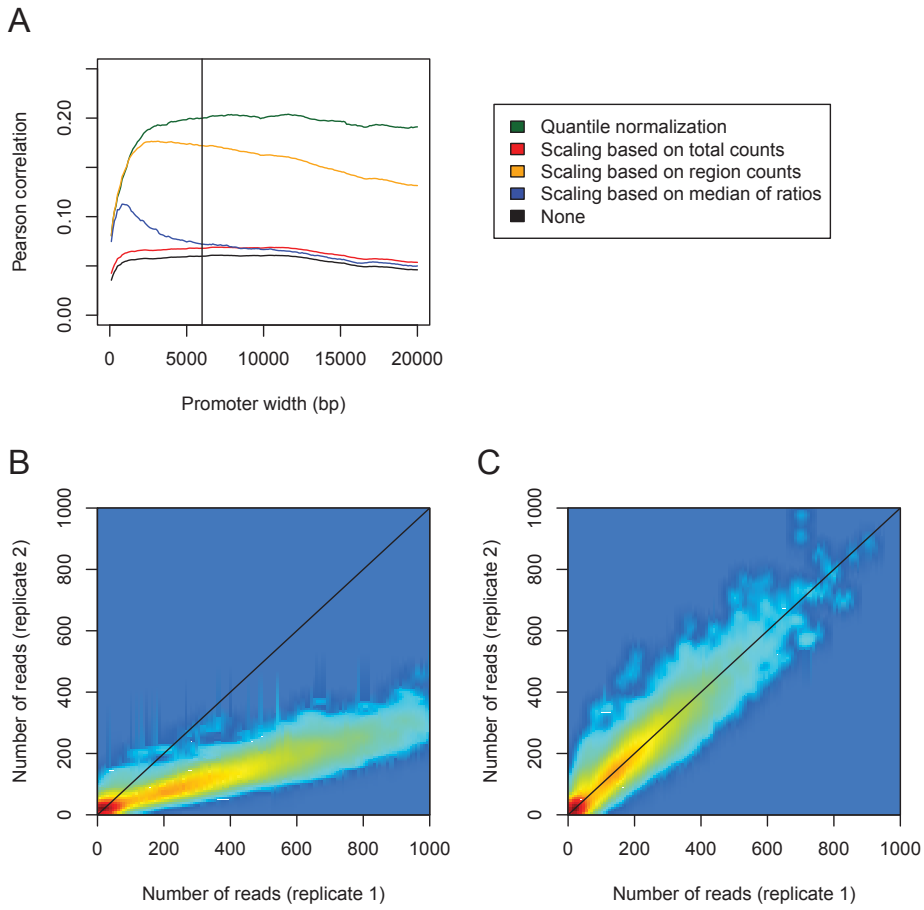

Figure A.16: Quantile normalization of ChIP-seq values in the *Cebpa* knockout data set. Figure **A** shows the Pearson correlation between differences in gene transcription and differences in ChIP-seq values for different promoter sizes (x-axis) and different ChIP-seq normalization approaches. The black vertical line indicates the promoter width of $\varpi = 6\,000$ chosen for this data set. Quantile normalization performs superior compared to linear scaling based on the total number of reads as used by many peak detection algorithms, e.g. MACS (Zhang *et al.*, 2008), or compared to scaling based on the median of count ratios as proposed by Anders and Huber, 2010, for RNA-seq data (see Section A.9 of the Appendix for details on the different normalization methods). A scatter plot (Figure **B**) between the number of reads observed within the promoters ($\varpi = 6\,000$) in the first and the second *Cebpa* knockout replicate sample shows larger read counts in the promoter regions for replicate 1, although replicate 2 had more mapped reads in total (Table A.12). This indicates different ChIP efficiency between both replicates and thus a need for normalization. Figure **C** shows the scatterplot after quantile normalization.

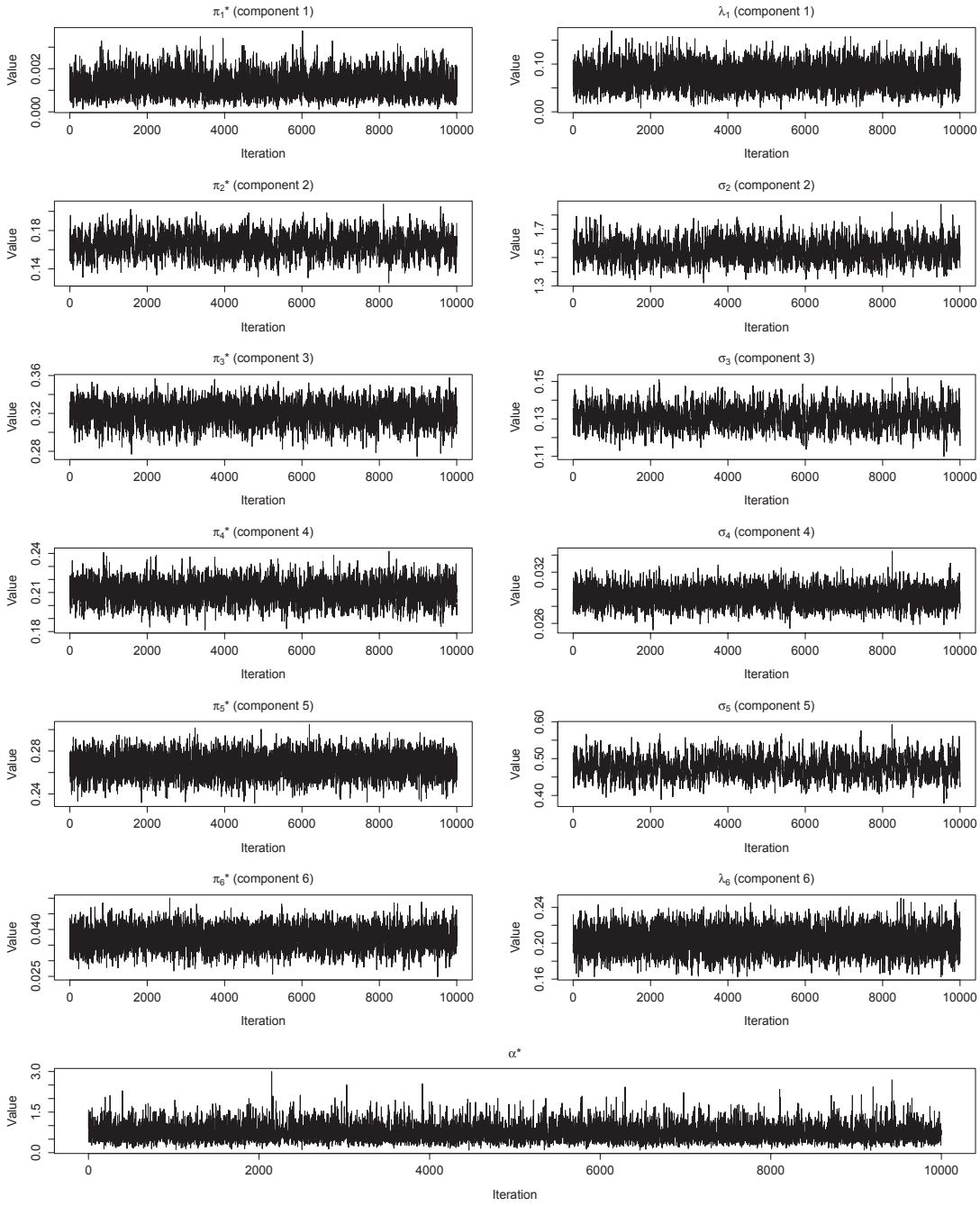

Figure A.17: Trace plots of the model parameters after fitting the Bayesian mixture model employing normal and exponential components (see Section 5.4.3) to the *Cebpa* knockout data set. The first 2 000 iterations are discarded and the remaining iterations are thinned by keeping every 10th iteration. The remaining 10 000 iterations are used for parameter estimation and are shown in the trace plots.

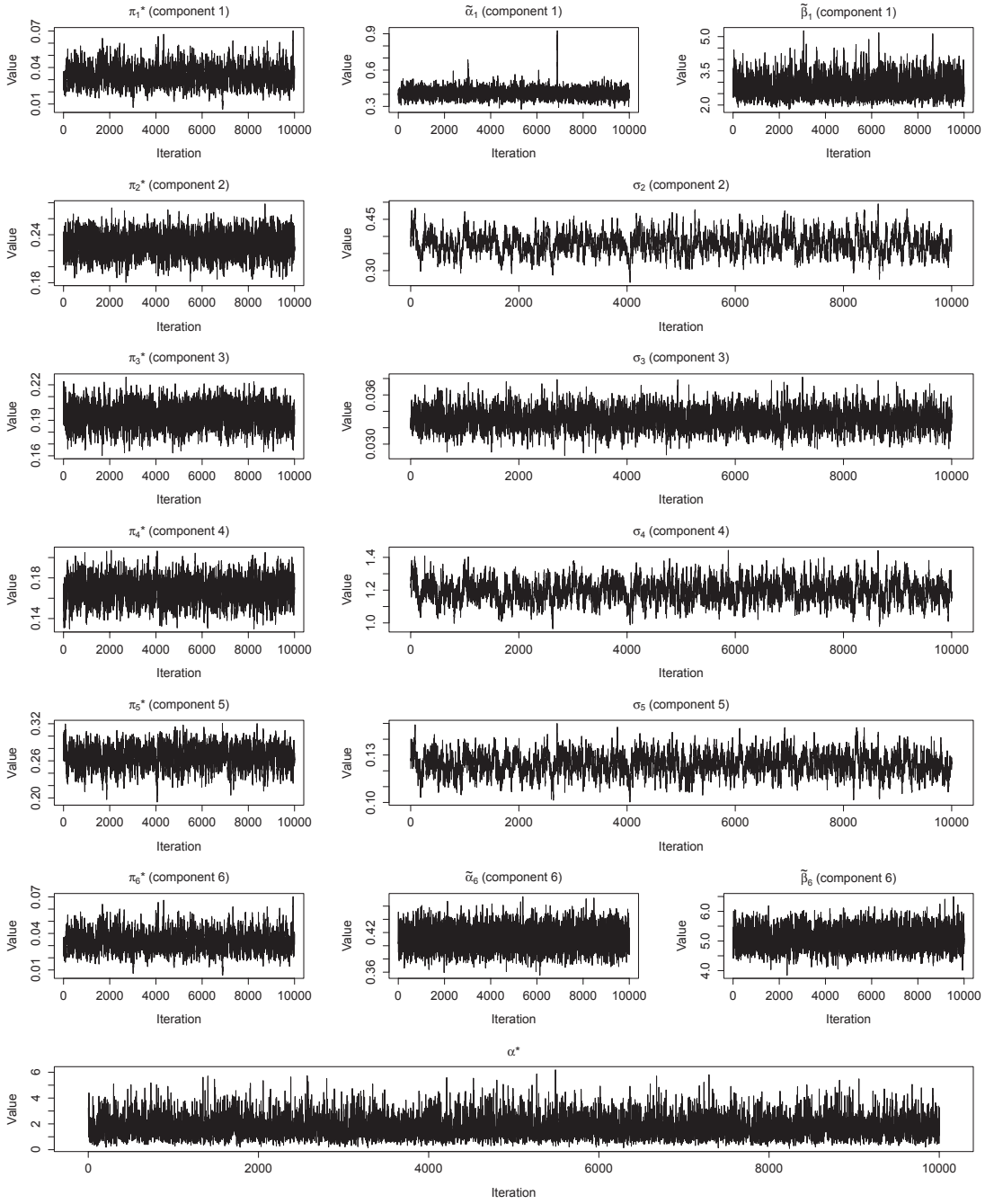

Figure A.18: Trace plots of the model parameters after fitting the Bayesian mixture model employing normal and gamma components (see Section 5.4.3) to the *Cebpa* knockout data set. The first 2 000 iterations are discarded and the remaining iterations are thinned by keeping every 10th iteration. The remaining 10 000 iterations are used for parameter estimation and are shown in the trace plots.

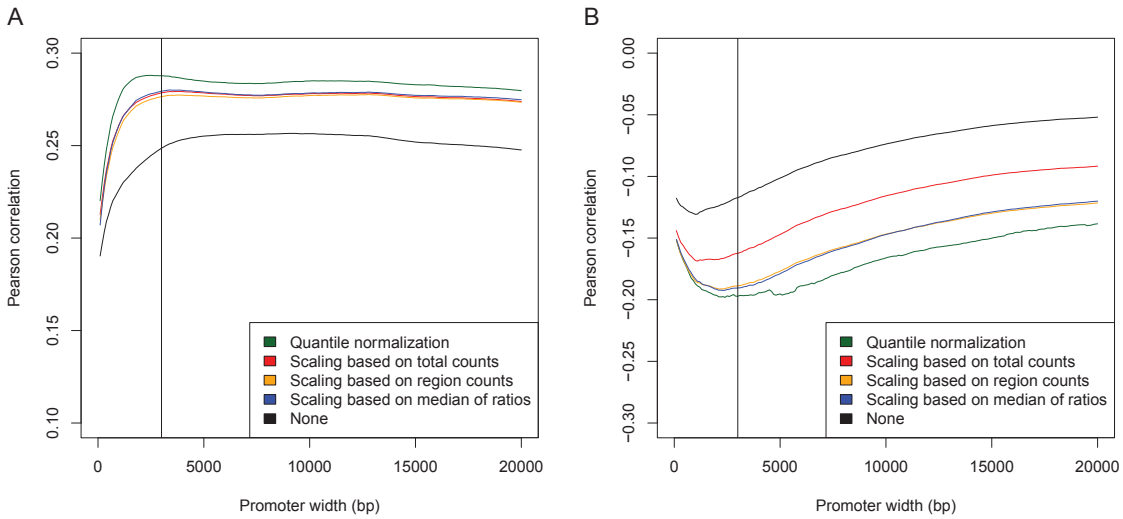

Figure A.19: Normalization of the ChIP-seq values from the prostate cancer data set. Pearson correlations between differences in transcription and differences in ChIP-seq values are calculated for different promoter sizes (x-axis) and different ChIP-seq normalization approaches. Figure **A** shows the correlation for the activating H3K4me3 mark. The black vertical line indicates the promoter width of $w = 3\,000$ chosen for this data set. Figure **B** shows the results for the repressive H3K27me3 mark, where the same promoter width of $w = 3\,000$ is chosen. Interestingly, although it is known that H3K27me3 occurs at the TSS but also throughout the gene body (Hebenstreit *et al.*, 2011; Ernst and Kellis, 2010), a larger window width w covering larger parts of the gene body does not lead to a stronger negative correlation. Recently, Dong *et al.* (2012) also chose windows close to the TSS to predict transcription based on the occurrence of H3K27me3. For both histone modifications, quantile normalization of the ChIP-seq values leads to the largest absolute correlation with the transcription data. The absolute correlation decreases slowly with increasing promoter widths w, suggesting that a broadly chosen promoter is not unfavorable for the subsequent analysis.

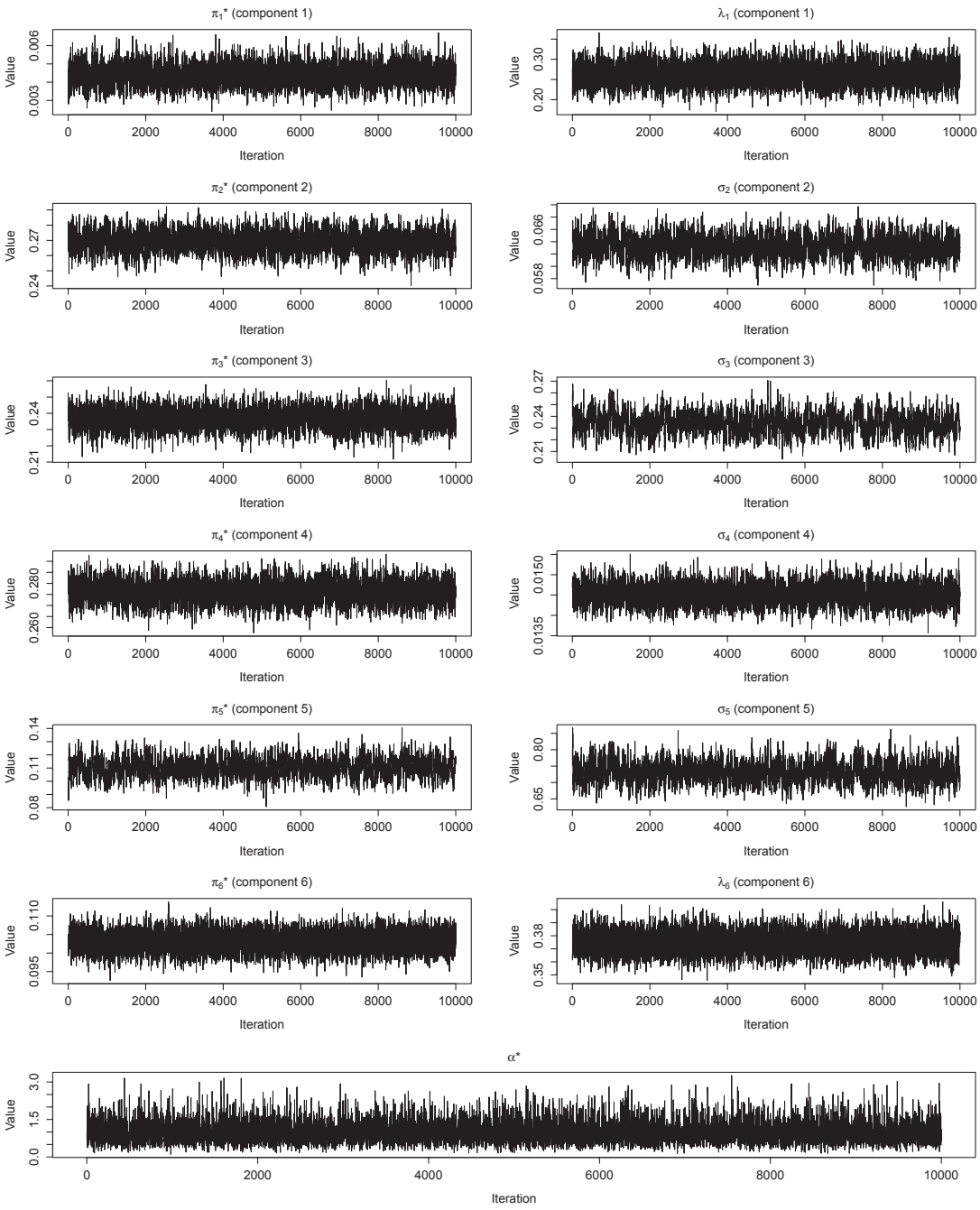

Figure A.20: Trace plots of the model parameters after fitting the Bayesian mixture model employing normal and exponential components (see Section 5.4.3) to the prostate cancer data set (H3K4me3). The first 2 000 iterations are discarded and the remaining iterations are thinned by keeping every 10th iteration. The remaining 10 000 iterations are used for parameter estimation and are shown in the trace plots.

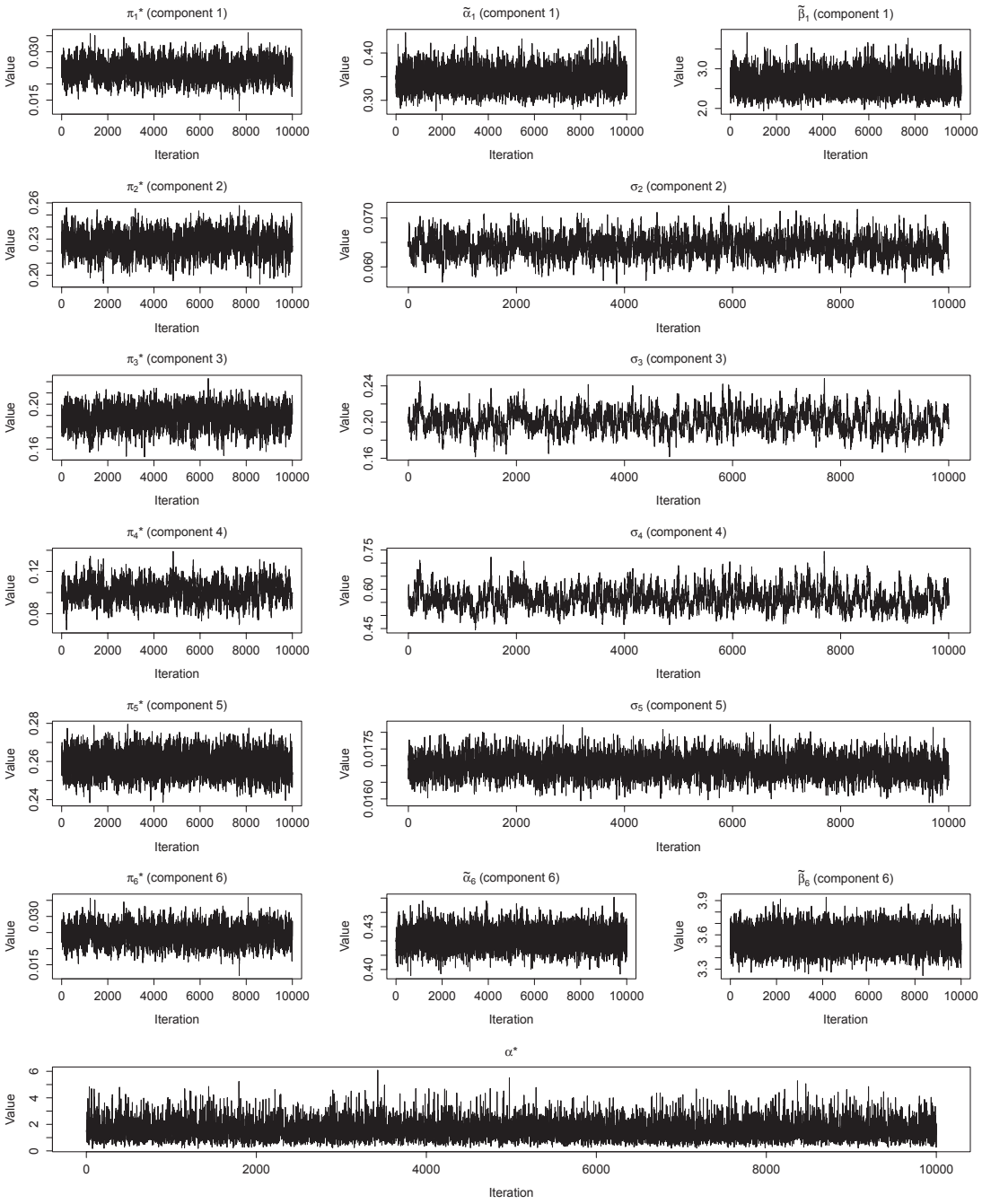

Figure A.21: Trace plots of the model parameters after fitting the Bayesian mixture model employing normal and gamma components (see Section 5.4.3) to the prostate cancer data set (H3K4me3). The first 2 000 iterations are discarded and the remaining iterations are thinned by keeping every 10th iteration. The remaining 10 000 iterations are used for parameter estimation and are shown in the trace plots.

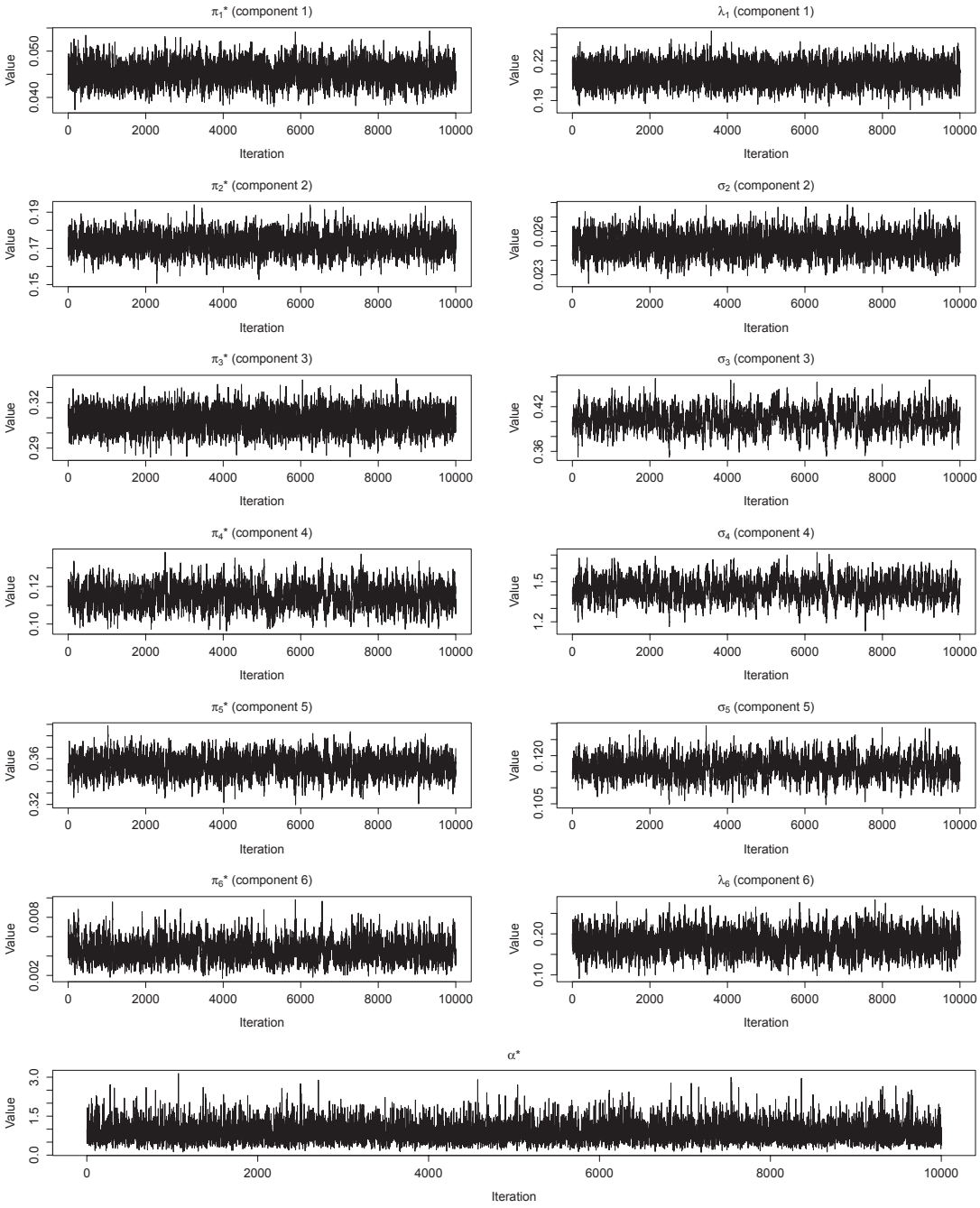

Figure A.22: Trace plots of the model parameters after fitting the Bayesian mixture model employing normal and exponential components (see Section 5.4.3) to the prostate cancer data set (H3K27me3). The first 2 000 iterations are discarded and the remaining iterations are thinned by keeping every 10th iteration. The remaining 10 000 iterations are used for parameter estimation and are shown in the trace plots.

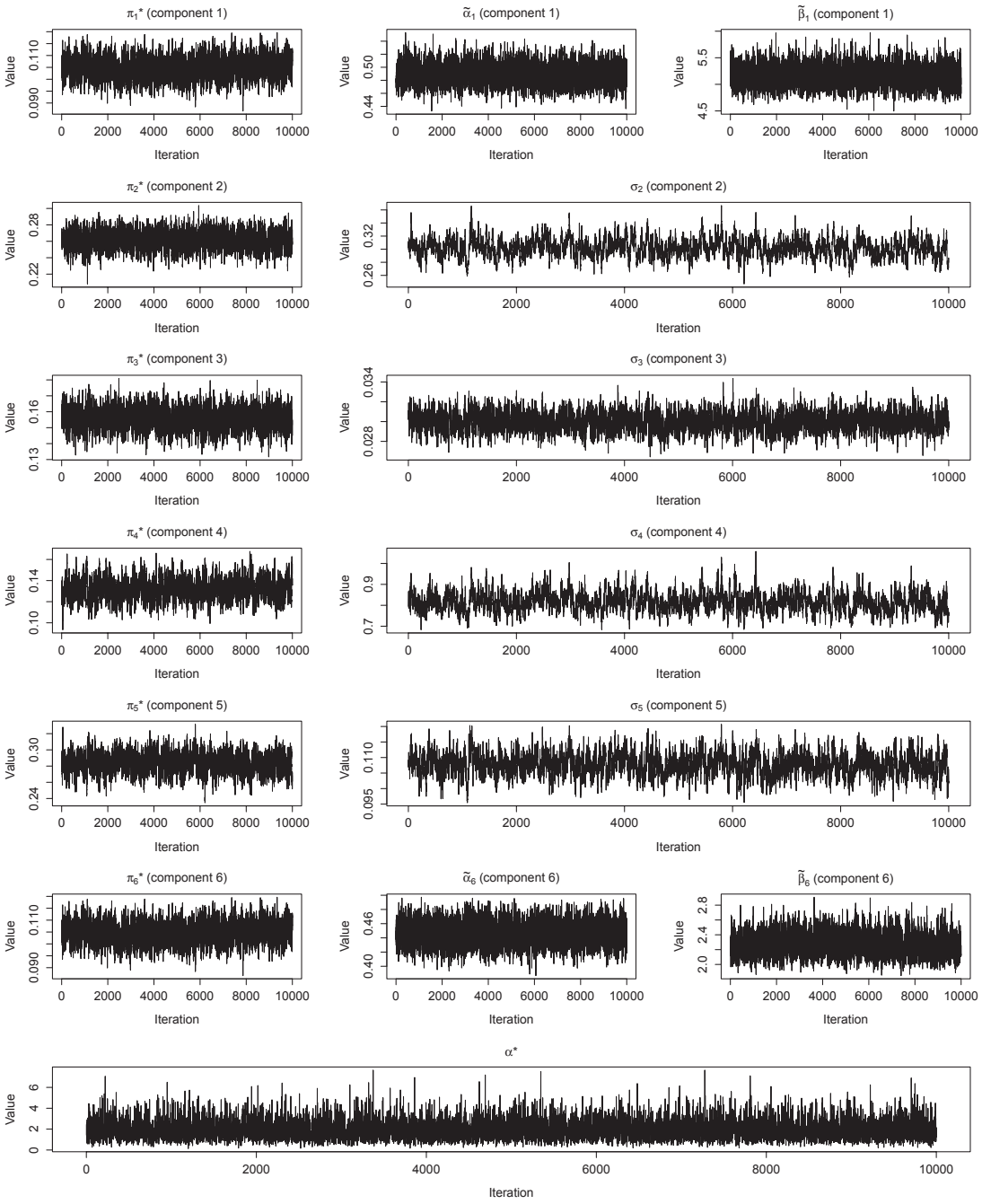

Figure A.23: Trace plots of the model parameters after fitting the Bayesian mixture model employing normal and gamma components (see Section 5.4.3) to the prostate cancer data set (H3K27me3). The first 2 000 iterations are discarded and the remaining iterations are thinned by keeping every 10th iteration. The remaining 10 000 iterations are used for parameter estimation and are shown in the trace plots.

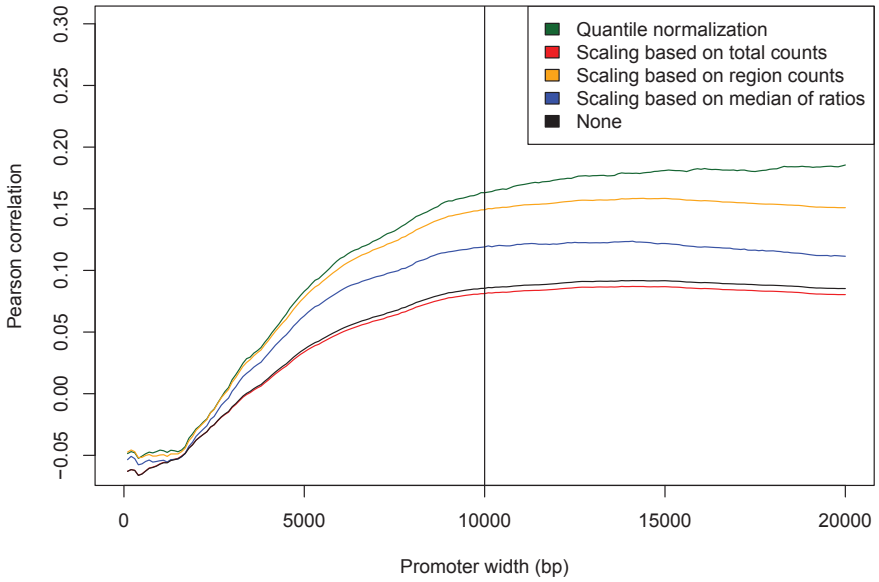

Figure A.24: Normalization of the ChIP-seq values from the ATRA and TCP treatment data set. Pearson correlations between differences in transcription values and differences in ChIP-seq values are calculated for different promoter sizes (x-axis) and different ChIP-seq normalization approaches. The black vertical line indicates the promoter width of $w = 10\,000$ chosen for this data set. The different normalization approaches are described in Section A.9 of the Appendix. The plot has to be interpreted carefully since it is known that H3K4me2 may occur at the promoters of active as well as inactive transcripts (Zhang *et al.*, 2009). Hence, some transcripts are supposed to show a positive correlation between H3K4me2 and gene transcription whereas others are supposed to show a negative one.

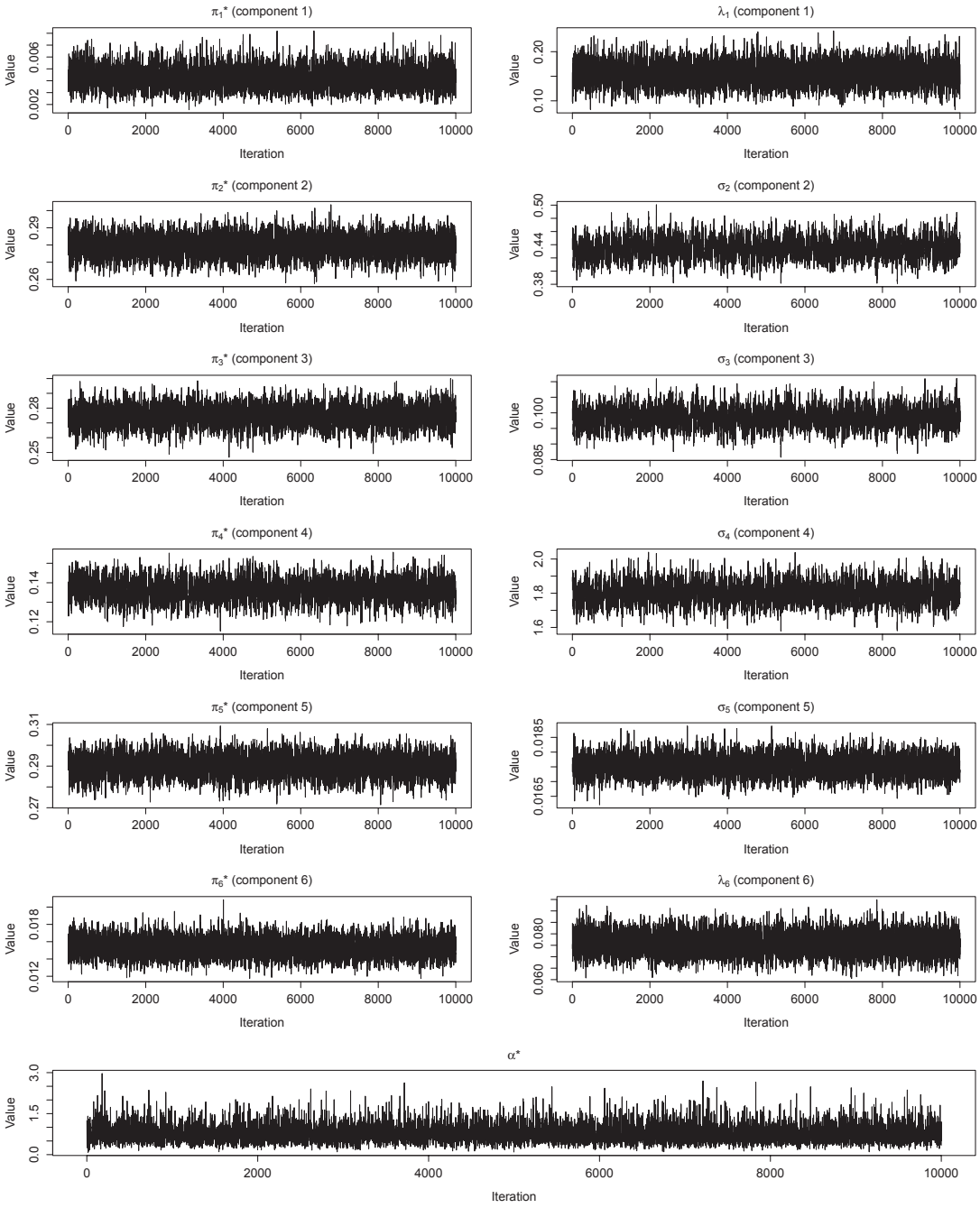

Figure A.25: Trace plots of the model parameters after fitting the Bayesian mixture model employing normal and exponential components (see Section 5.4.3) to the ATRA and TCP treatment data set. The first 2 000 iterations are discarded and the remaining iterations are thinned by keeping every 10th iteration. The remaining 10 000 iterations were used for parameter estimation and are shown in the trace plots.

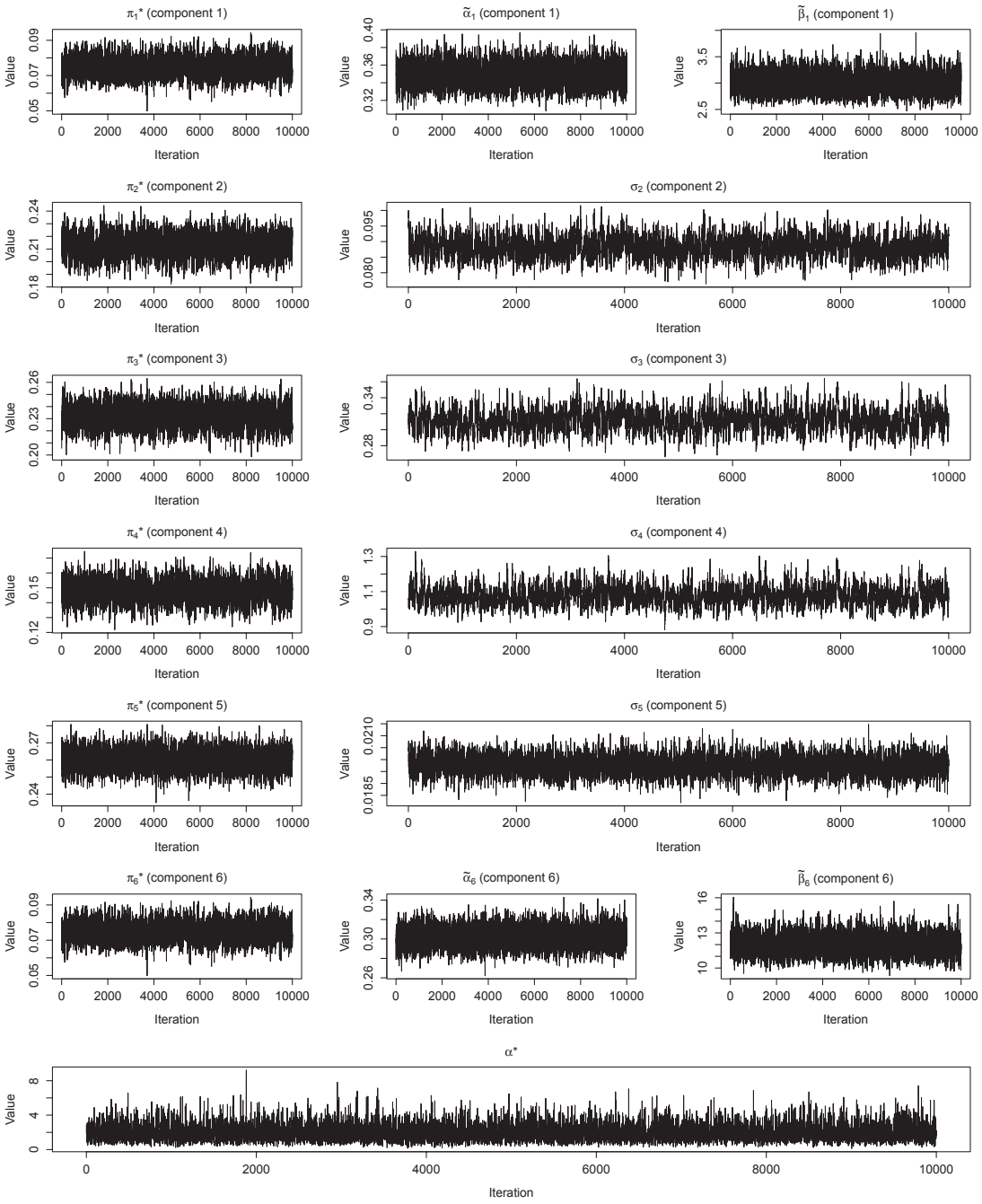

Figure A.26: Trace plots of the model parameters after fitting the Bayesian mixture model employing normal and gamma components (see Section 5.4.3) to the ATRA and TCP treatment data set. The first 2 000 iterations are discarded and the remaining iterations are thinned by keeping every 10th iteration. The remaining 10 000 iterations are used for parameter estimation and are shown in the trace plots.

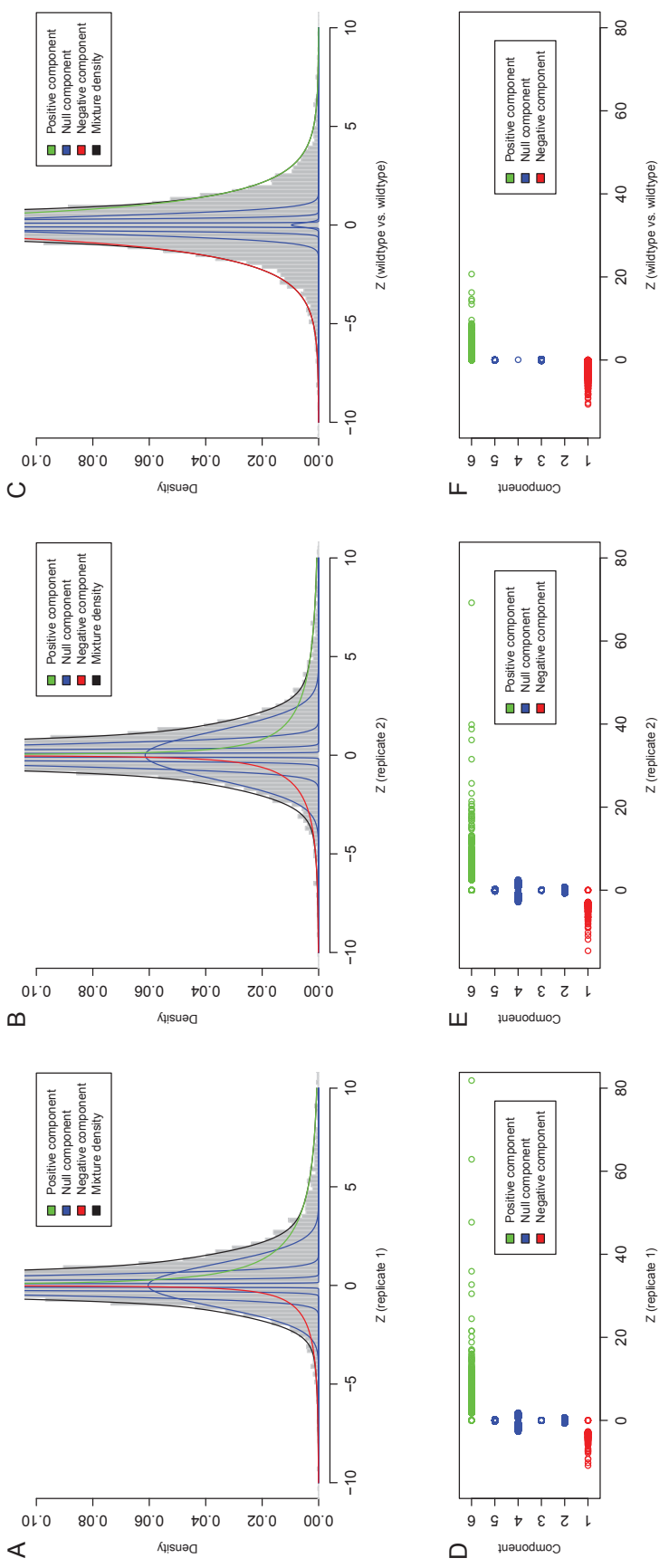

Figure A.27: Model fits and classifications for subsets of the *Cebpa* knockout data set, employing a mix of normal and gamma distributions. Figure **A** (**B**) shows the observed z_i values and the mixture model's fit when comparing the first (second) *Cebpa* knockout replicate to the first (second) wild-type replicate. The corresponding classifications are given in Figure **D** and **E**, respectively. Figures **C** and **F** present the mixture model of the comparison of the first to the second *Cebpa* knockout replicate.

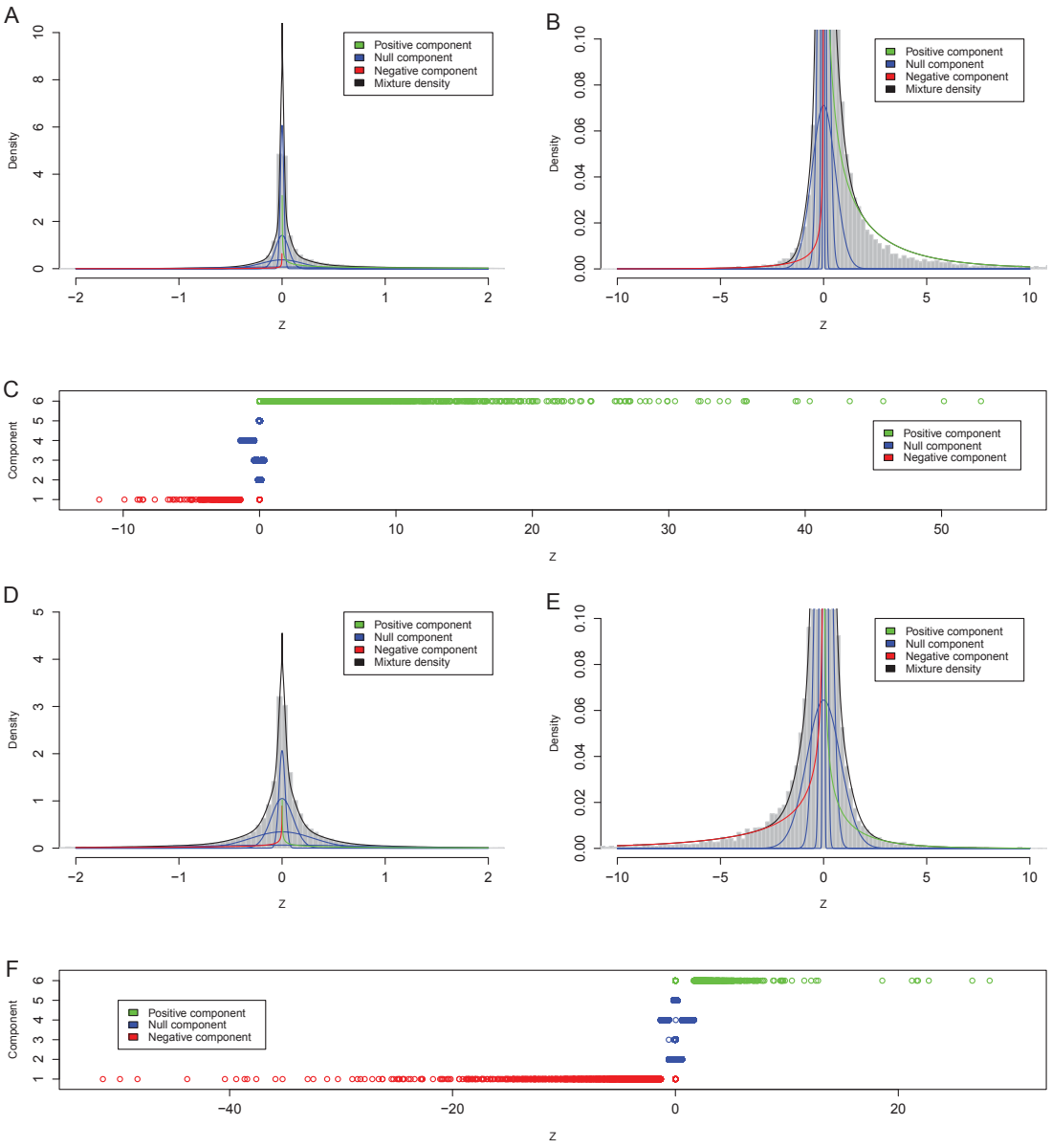

Figure A.28: Model fit and classification for the prostate cancer data set, employing a mix of normal and gamma distributions. The two grey histograms **A** and **B** show the distribution of the observed z_i values at different scales after matching the transcription data to the H3K4me3 ChIP-seq data. The histograms are overlaid with the mixture density and the densities of the single components. In Figure **C**, the observed z_i are plotted against their classification. Figures **D**, **E** and **F** are the corresponding plots when using the ChIP-seq data for the H3K27me3 histone modification.

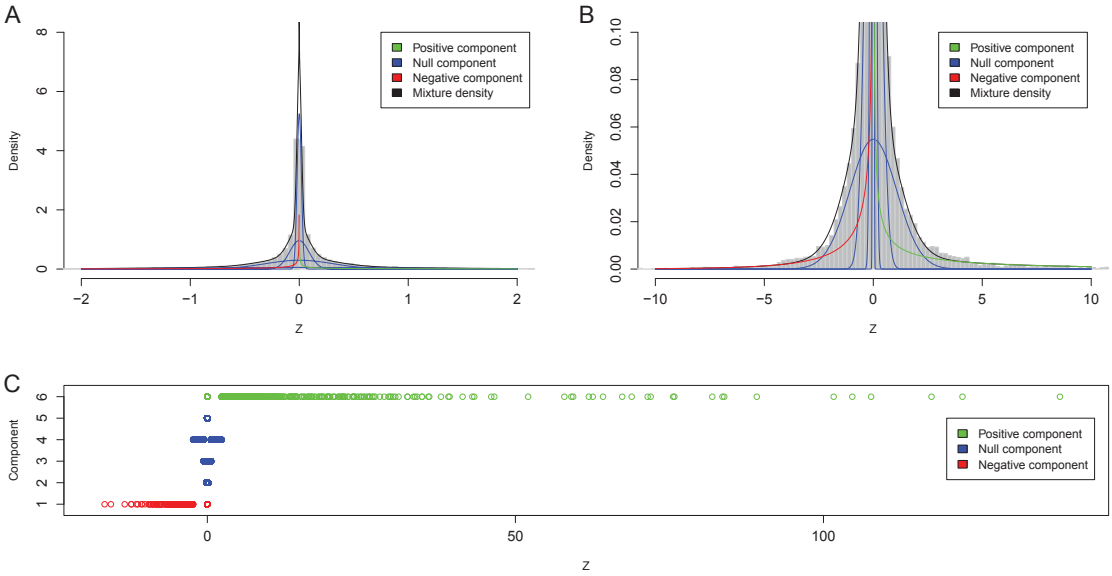

Figure A.29: Model fit and classification for the ATRA and TCP treatment data set, employing a mix of normal and gamma distributions. The two grey histograms **A** and **B** show the distribution of the observed z_i values at different scales. The histograms are overlaid with the mixture density and the densities of the single components. In Figure **C**, the observed z_i are plotted against their classification.

Sample	Mapped reads	Uniquely mapped reads	Uniquely mapped unique reads
Knockout replicate 1	9 993 417	8 664 411	7 571 951
Knockout replicate 2	11 349 598	9 408 692	8 352 119
Wild-type replicate 1	7 968 878	6 847 932	5 523 503
Wild-type replicate 2	15 926 036	13 218 019	11 883 365

Table A.12: Alignment statistics of the *Cebpa* knockout data set. DNA was fragmented into $\sim$ 200 bp fragments. The first 36 bases starting at the 5 prime end were sequenced. Reads mapping to more than one position as well as duplicated reads were discarded. The numbers of remaining reads used in the presented analysis are given in the last column.

Sample	Mapped reads	Uniquely mapped reads	Uniquely mapped unique reads
LNCaP H3K4me3	23 344 847	20 934 293	17 929 262
PrEC H3K4me3	18 693 026	19 999 178	14 309 693
LNCaP H3K27me3	50 906 313	44 459 537	36 273 090
PrEC H3K27me3	31 950 793	27 890 158	21 901 684
LNCaP RNA-seq	28 959 521	22 595 448	8 735 867
PrEC RNA-seq	26 972 636	21 646 919	7 747 389

Table A.13: Alignment statistics of the prostate cancer data set. DNA was fragmented into $\sim$ 350 bp fragments for ChIP-seq experiments. The first 49 (50) bases starting at the 5 prime end were sequenced after ChIP against H3K4me3 (H3K27me). Reads mapping to more than one position as well as duplicated reads were discarded. The numbers of remaining reads used in the presented analysis are given in the last column. For RNA-seq experiments, sequencing length was 76 bp and all mapped reads were used to estimate transcripts' FPKM values using the Cufflinks algorithm (Trapnell *et al.*, 2010).

Sample	Mapped reads	Uniquely mapped reads	Uniquely mapped unique reads
ATRA + TCP	40 346 902	37 534 898	31 880 435
ATRA	39 386 754	36 860 942	31 569 702

Table A.14: Alignment statistics of the ATRA and TCP treatment data set. DNA was fragmented into $\sim$ 200 bp fragments. The first 58 bases starting at the 5 prime end were sequenced. Reads mapping to more than one position as well as duplicated reads were discarded. The numbers of remaining reads used in the presented analysis are given in the last column.

	Negative	Null				Positive
k	1	2	3	4	5	6
$\hat{\sigma}_k^2$	-	0.004	0.055	0	0.535	-
	-	[0.004,0.004]	[0.047,0.063]	[0,0]	[0.45,0.63]	-
$\hat{\lambda}_k$	0.262	-	-	-	-	0.375
	[0.213,0.314]	-	-	-	-	[0.359,0.391]
$\hat{\pi}_k^*$	0.004	0.269	0.237	0.276	0.11	0.103
	[0.003,0.006]	[0.256,0.282]	[0.225,0.248]	[0.267,0.286]	[0.096,0.125]	[0.098,0.108]
		0.893 [0.887, 0.898]				
Number of genes	137	12 568	10 204	16 719	3 450	3 579
		42 941				
Proportion	0.003	0.269	0.219	0.358	0.074	0.077
		0.920				

Table A.15: Estimated parameters of the Bayesian mixture model for the prostate cancer data set (H3K4me3), employing a mix of normal and exponential distributions. The 95% credible intervals are given in square brackets. The curly brackets summarize the weights π_k^*, the number of classified genes or the proportion of classified genes for all four null components, respectively. The acceptance rate for the Metropolis-Hastings step for α^* in the MCMC run is 0.4678.

	Negative	Null				Positive
k	1	2	3	4	5	6
$\hat{\sigma}_k^2$	-	0.001	0.163	2.09	0.014	-
	-	[0.001,0.001]	[0.14,0.185]	[1.671,2.535]	[0.012,0.015]	-
$\hat{\lambda}_k$	0.21	-	-	-	-	0.179
	[0.196,0.225]	-	-	-	-	[0.129,0.233]
$\hat{\pi}_k^*$	0.045	0.173	0.308	0.115	0.354	0.005
	[0.041,0.05]	[0.162,0.183]	[0.295,0.322]	[0.104,0.127]	[0.337,0.37]	[0.003,0.007]
		0.95 [0.944, 0.956]				
Number of genes	1 328	11 142	12 035	3 830	18 209	113
		45 216				
Proportion	0.028	0.239	0.258	0.082	0.390	0.002
		0.969				

Table A.16: Estimated parameters of the Bayesian mixture model for prostate cancer data set (H3K27me3), employing a mix of normal and exponential distributions. The 95% credible intervals are given in square brackets. The curly brackets summarize the weights π_k^*, the number of classified genes or the proportion of classified genes for all four null components, respectively. The acceptance rate for the Metropolis-Hastings step for α^* in the MCMC run is 0.4338.

	Negative	Null				Positive
k	1	2	3	4	5	6
$\hat{\sigma}_k^2$	-	0.189	0.01	3.247	0	-
	-	[0.164,0.218]	[0.008,0.011]	[2.835,3.707]	[0,0]	-
$\hat{\lambda}_k$	0.154	-	-	-	-	0.076
	[0.113,0.199]	-	-	-	-	[0.067,0.085]
$\hat{\pi}_k^*$	0.004	0.279	0.275	0.136	0.29	0.015
	[0.003,0.006]	[0.268,0.291]	[0.261,0.288]	[0.125,0.147]	[0.28,0.3]	[0.013,0.017]
		0.98 [0.977, 0.983]				
Number of genes	64	7 926	8 842	3 179	11 393	349
		31 340				
Proportion	0.002	0.250	0.278	0.100	0.359	0.011
		0.987				

Table A.17: Estimated parameters of the Bayesian mixture model for the ATRA and TCP treatment data set, employing a mix of normal and exponential distributions. The 95% credible intervals are given in square brackets. The curly brackets summarize the weights π_k^*, the number of classified genes or the proportion of classified genes for all four null components, respectively. The acceptance rate for the Metropolis-Hastings step for α^* in the MCMC run is 0.4041.

	Negative	Null				Positive
k	1	2	3	4	5	6
$\hat{\sigma}_k^2$	-	0.004	0.041	0.320	0	-
	-	[0.004,0.005]	[0.032,0.051]	[0.246,0.422]	[0,0]	-
$\hat{\tilde{\alpha}}_k$	0.342	-	-	-	-	0.422
	[0.306,0.385]	-	-	-	-	[0.409,0.435]
$\hat{\tilde{\beta}}_k$	2.625	-	-	-	-	3.557
	[2.205,3.151]	-	-	-	-	[3.387,3.734]
$\hat{\pi}_k^*$	0.024	0.226	0.188	0.101	0.259	0.202
	[0.019,0.030]	[0.209,0.243]	[0.170,0.205]	[0.083,0.119]	[0.248,0.270]	[0.194,0.210]
				0.774		
				[0.762, 0.786]		
Number of genes	331	11 195	8 328	1 278	17 108	7 698
				37 909		
Proportion	0.007	0.244	0.181	0.028	0.372	0.168
				0.825		

Table A.18: Estimated parameters of the Bayesian mixture model for the prostate cancer data set (H3K4me3), employing a mix of normal and gamma distributions. The 95% credible intervals are given in square brackets. The curly brackets summarize the weights π_k^*, the number of classified genes or the proportion of classified genes for all four null components, respectively. The acceptance rates for the Metropolis-Hastings steps for α^*, $a_{\tilde{\alpha}_k}$ and $c_{\tilde{\beta}_k}$ in the MCMC run are 0.551, 0.355 and 0.156, respectively.

	Negative	Null				Positive
k	1	2	3	4	5	6
$\hat{\sigma}_k^2$	-	0.091	0.001	0.668	0.012	-
	-	[0.075,0.11]	[0.001,0.001]	[0.533,0.839]	[0.01,0.013]	-
$\hat{\tilde{\alpha}}_k$	0.489	-	-	-	-	0.445
	[0.46,0.521]	-	-	-	-	[0.416,0.477]
$\hat{\tilde{\beta}}_k$	5.159	-	-	-	-	2.265
	[4.815,5.536]	-	-	-	-	[2.012,2.567]
$\hat{\pi}_k^*$	0.105	0.261	0.156	0.132	0.284	0.063
	[0.096,0.113]	[0.24,0.282]	[0.143,0.168]	[0.113,0.152]	[0.26,0.306]	[0.053,0.073]
		0.832 [0.816, 0.850]				
Number of genes	2 692	11 414	11 269	4 368	15 657	740
		42 708				
Proportion	0.058	0.247	0.244	0.095	0.339	0.016
		0.926				

Table A.19: Estimated parameters of the Bayesian mixture model for prostate cancer data set (H3K27me3), employing a mix of normal and gamma distributions. The 95% credible intervals are given in square brackets. The curly brackets summarize the weights π_k^*, the number of classified genes or the proportion of classified genes for all four null components, respectively. The acceptance rates for the Metropolis-Hastings steps for α^*, $a_{\tilde{\alpha}_k}$ and $c_{\tilde{\beta}_k}$ in the MCMC run are 0.587, 0.242 and 0.287, respectively.

	Negative	Null				Positive
k	1	2	3	4	5	6
$\hat{\sigma}_k^2$	-	0.008	0.097	1.159	0	-
	-	[0.007,0.009]	[0.081,0.116]	[0.953,1.429]	[0,0]	-
$\hat{\tilde{\alpha}}_k$	0.349	-	-	-	-	0.302
	[0.325,0.373]	-	-	-	-	[0.283,0.321]
$\hat{\tilde{\beta}}_k$	2.994	-	-	-	-	11.893
	[2.678,3.367]	-	-	-	-	[10.462,13.607]
$\hat{\pi}_k^*$	0.075	0.213	0.232	0.148	0.26	0.072
	[0.065,0.085]	[0.196,0.229]	[0.215,0.249]	[0.134,0.162]	[0.249,0.271]	[0.063,0.081]
				0.853		
				[0.835, 0.870]		
Number of genes	540	7 131	7 437	3 957	11 377	909
				29 902		
Proportion	0.017	0.227	0.237	0.126	0.363	0.029
				0.954		

Table A.20: Estimated parameters of the Bayesian mixture model for the ATRA and TCP treatment data set, employing a mix of normal and gamma distributions. The 95% credible intervals are given in square brackets. The curly brackets summarize the weights π_k^*, the number of classified genes or the proportion of classified genes for all four null components, respectively. The acceptance rates for the Metropolis-Hastings steps for α^*, $a_{\tilde{\alpha}_k}$ and $c_{\tilde{\beta}_k}$ in the MCMC run are 0.600, 0.258 and 0.234, respectively.

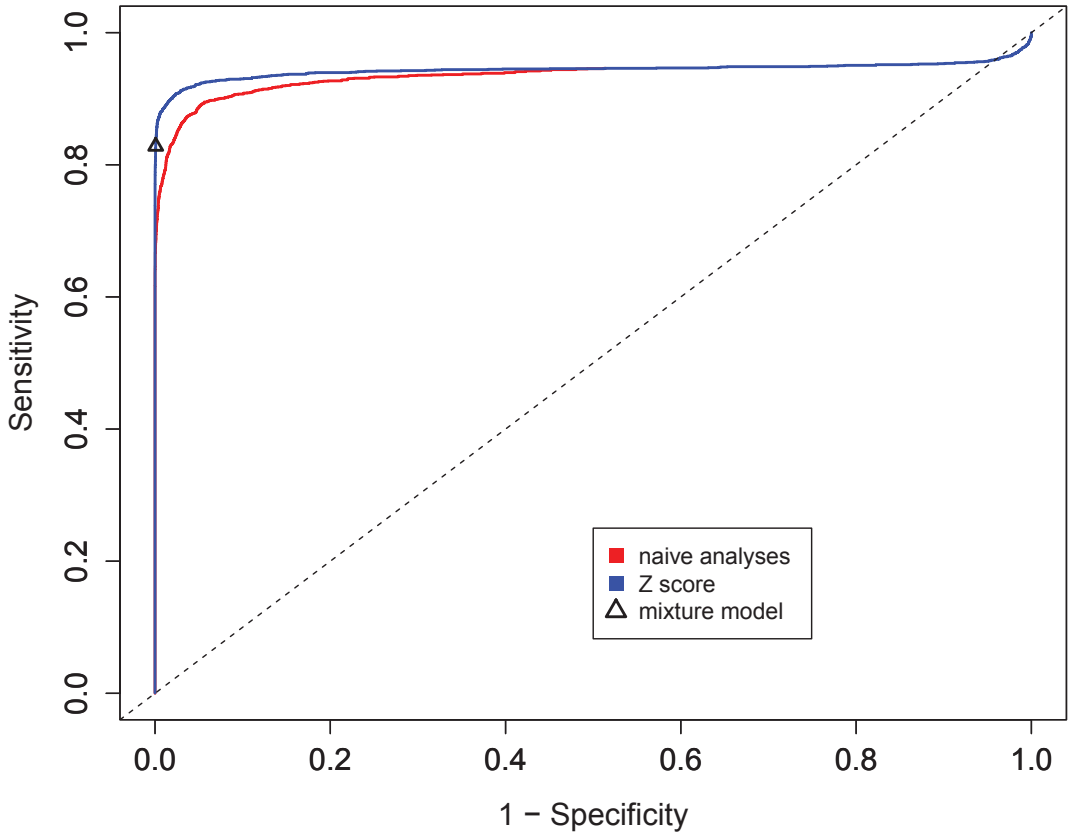

Figure A.30: ROC based comparison of the Z score (blue line) to a naive approach (red line) using simulated data, employing a mix of normal and exponential distributions. The mixture model (black triangle) classifies 1 651 transcripts with differences in both data sets correctly to the positive component. 349 differential transcripts are falsely classified to a null or negative component resulting in a sensitivity of 0.826. The observed specificity was 0.999. The red curve represents the performance of naive separate analyses of both data types as described in Section 5.4.7. At the same level of specificity, the naive approach achieves a lower sensitivity of 0.6825.

Appendix B

Additional Theoretical Considerations

Please also refer to Sections 4.1 and 4.2 for notation and definitions used in the following.

Theorem B.1 (**Key quantities for sampling from double Matérn cluster process**). *The formulas are derived in two steps: at first for fixed cluster sizes S_c and fixed metacluster sizes S_{mc}, while in a second step, random cluster sizes are permitted and a Taylor expansion is used to obtain formulas for the corresponding expected cluster size μ_c and expected metacluster size μ_{mc}.*

Part 1: Sampling with fixed cluster sizes. *When a proportion of*

$$\tilde{p}^*_{mc} = \begin{cases} \frac{1}{\frac{S_{mc}}{p_{mc}} - S_{mc} + 1} & \text{if } p_{mc} \neq 0, \\ 0 & \text{if } p_{mc} = 0, \end{cases}$$

of first generation points is replaced by a metacluster containing S_{mc} second generation points, and a proportion of

$$\tilde{p}^*_{c} = \begin{cases} \frac{1}{\frac{S_{c}}{p_{c}} - S_{c} + 1} & \text{if } p_{c} \neq 0, \\ 0 & \text{if } p_{c} = 0, \end{cases}$$

of all the existent points (first and second generation points) is replaced by a

cluster of S_c third generation points, then the overall proportion of clustered points will be approximately p_c. Starting with

$$\begin{aligned} n_\eta &= \frac{n_{total}}{\tilde{p}^*_{mc} \cdot S_{mc}\Big[\tilde{p}^*_c \cdot S_c + (1-\tilde{p}^*_c)\Big] + (1-\tilde{p}^*_{mc})\Big[\tilde{p}^*_c \cdot S_c + (1-\tilde{p}^*_c)\Big]} \\ &= \frac{n_{total}}{(\tilde{p}^*_c \cdot S_c + 1 - \tilde{p}^*_c) \cdot (\tilde{p}^*_{mc} \cdot S_{mc} + 1 - \tilde{p}^*_{mc})} \\ &= \frac{n_{total}}{\Big[\tilde{p}^*_c(S_c - 1) + 1\Big] \cdot \Big[\tilde{p}^*_{mc}(S_{mc} - 1) + 1\Big]} \end{aligned}$$

first generation points, one obtains a point pattern consisting of a total of approximately n_{total} points.

Part 2: Sampling with random cluster sizes. *When a proportion of*

$$\tilde{p}_{mc} = \begin{cases} \frac{1}{\frac{\mu_{mc}}{p_{mc}} - \mu_{mc} + 1} & \text{if } p_{mc} \neq 0, \\ 0 & \text{if } p_{mc} = 0, \end{cases} \tag{B.1}$$

of first generation points is replaced by a metacluster containing S_{mc} second generation points, where $S_{mc} \sim Poi(\mu_{mc})$, and a proportion of

$$\tilde{p}_c = \begin{cases} \frac{1}{\frac{\mu_c}{p_c} - \mu_c + 1} & \text{if } p_c \neq 0, \\ 0 & \text{if } p_c = 0, \end{cases} \tag{B.2}$$

of all the existent points (first and second generation points) is replaced by a cluster of S_c third generation points, where $S_c \sim Poi(\mu_c)$, then the overall proportion of clustered points will be approximately p_c. Starting with

$$n_\eta = \frac{n_{total}}{\Big[\tilde{p}_c(\mu_c - 1) + 1\Big] \cdot \Big[\tilde{p}_{mc}(\mu_{mc} - 1) + 1\Big] \cdot \Big[1 - \frac{1-p_c}{p_c} \cdot \tilde{p}^2_c - \frac{1-p_{mc}}{p_{mc}} \cdot \tilde{p}^2_{mc}\Big]}$$

first generation points, one obtains a point pattern consisting of a total of approximately n_{total} points.

Proof. **Part 1:** Consider a point pattern consisting of n_{total} points in which $p_{mc} \cdot n_{total} = S_{mc}$, i.e. where n_{total} is such that the number $p_{mc} \cdot n_{total}$ of points in metaclusters is equal to the size of a metacluster, resulting in a point pattern with exactly one metacluster. Then

$$n_{total} - p_{mc} \cdot n_{total} + 1 = \frac{S_{mc}}{p_{mc}} - S_{mc} + 1 \tag{B.3}$$

corresponds to the number of first generation points. Since there will only be one metacluster, the reciprocal value $\tilde{p}^*_{mc}$ is equal to the proportion of first generation points points that is to be replaced by second generation points. Note that (B.3) does not contain S_c and p_c since by construction, points in and outside metaclusters have the same probability to be replaced by clusters, and, thus, the clusters do not matter here.

To derive the formula for $\tilde{p}^*_c$, one can neglect the metaclustering since for potential replacement of a point by a cluster it is irrelevant by construction whether it is within a metacluster or not. In analogy to the argument for deriving $\tilde{p}^*_{mc}$, consider a point pattern consisting of n_{total} points in which $p_c \cdot n_{total} = S_c$, i.e. where n_{total} is such that the number of points $p_c \cdot n_{total}$ is equal to the size of a cluster, resulting in a point pattern with exactly one cluster. Then

$$n_{total} - p_c \cdot n_{total} + 1 = \frac{S_c}{p_c} - S_c + 1$$

corresponds to the total number of (first and second generation) points. Since there is only one cluster, the reciprocal value $\tilde{p}^*_c$ is equal to the proportion of points that is to be replaced by third generation points.

The restrictions made for n_{total} are without loss of generality since the claim does not contain n_{total} and only proportions are considered. However, the given argumentation implies that $p_{mc} \cdot n_{total} \in \mathbb{N}$. Since for many given values of n_{total} and p_{mc}, this will not be the case, the result is only approximate in general.

Number of first generation points: before the clusters are generated, there are $n_\eta \cdot \tilde{p}^*_{mc} \cdot S_{mc}$ points in a metacluster and $n_\eta \cdot (1 - \tilde{p}^*_{mc})$ points outside the metaclusters. Of all these points, a proportion of $\tilde{p}^*_c$ is replaced by clusters consisting of S_c points. This means that

$$\begin{aligned} n_{total} &= \Big[n_\eta \cdot \tilde{p}^*_{mc} \cdot S_{mc} + n_\eta \cdot (1 - \tilde{p}^*_{mc})\Big]\Big[\tilde{p}^*_c \cdot S_c + (1 - \tilde{p}^*_c)\Big] \\ &= n_\eta \cdot \Big[\tilde{p}^*_{mc} \cdot S_{mc} + (1 - \tilde{p}^*_{mc})\Big]\Big[\tilde{p}^*_c \cdot S_c + (1 - \tilde{p}^*_c)\Big]. \end{aligned}$$

Solving for n_η gives the result. Please note that the discussed potential inaccuracies arising in the derivation of $\tilde{p}^*_c$ and $\tilde{p}^*_{mc}$ carry through to the calculation of n_{total} here.

Part 2: If the cluster size is no longer fixed, but random, to calculate $\tilde{p}_{mc}$ and $\tilde{p}_c$ it is sufficient to follow the argumentation in part 1) of the proof and to replace S_c by μ_c and S_{mc} by μ_{mc} in the formulas (B.1) and (B.2): if there is only one metacluster, its size automatically will be equal to the mean metacluster size μ_{mc}, and if there is only one cluster, its size automatically will be equal to the mean cluster size μ_c.

For the case of random cluster sizes and random metacluster sizes, one can approximate n_η by the law of error propagation, doing a second order Taylor expansion in two dimensions, corresponding to the random variables S_c and S_{mc}:

$$\begin{aligned} E[g(S_c, S_{mc})] \;\approx\; & g\Big(E(S_c), E(S_{mc})\Big) \\ & + \frac{1}{2}\Bigg(g_{SS}\Big(E(S_c), E(S_{mc})\Big) \cdot Var(S_c) \\ & + g_{S_{mc}S_{mc}}\Big(E(S_c), E(S_{mc})\Big) \cdot Var(S_{mc})\Bigg), \end{aligned}$$

where g_{xx} is the second order partial derivative of g with respect to x. The terms involving the first order partial derivatives disappear because $E\Big(S_c - E(S_c)\Big) = 0$

(this holds for any random variable), the term involving g_{SS^*} disappears because $Cov(S_c, S_{mc}) = 0$. The variable one wants to obtain the correct expectation for is n_{total}, thus, the expansion is done for

$$\begin{aligned} g(S_c, S_{mc}) &= n_\eta \cdot \left(\tilde{p}_c^*(S_c - 1) + 1\right) \cdot \left(\tilde{p}_{mc}^*(S_{mc} - 1) + 1\right) \\ &= n_\eta \cdot \left(\frac{S_c - 1}{\frac{S_c}{p_c} - S_c + 1} + 1\right) \cdot \left(\frac{S_{mc} - 1}{\frac{S_{mc}}{p_{mc}} - S_{mc} + 1} + 1\right), \end{aligned}$$

and after several reformulations and calculations one obtains

$$\begin{aligned} g_{S_c}(S_c, S_{mc}) &= \frac{n_\eta}{p_c} \cdot \left(\frac{S_{mc} - 1}{\frac{S_{mc}}{p_{mc}} - S_{mc} + 1} + 1\right) \cdot \frac{1}{\left(\frac{S_c}{p_c} - S_c + 1\right)^2} \\ &= n_\eta \cdot \frac{\tilde{p}_c^{*2}}{p_c} \cdot \left(\tilde{p}_{mc}^*(S_{mc} - 1) + 1\right), \\ g_{S_{mc}}(S_c, S_{mc}) &= \frac{n_\eta}{p_{mc}} \cdot \left(\frac{S_c - 1}{\frac{S_c}{p_c} - S_c + 1} + 1\right) \cdot \frac{1}{\left(\frac{S_{mc}}{p_{mc}} - S_{mc} + 1\right)^2} \\ &= n_\eta \cdot \frac{\tilde{p}_{mc}^{*\,2}}{p_{mc}} \cdot \left(\tilde{p}_c^*(S_c - 1) + 1\right), \\ g_{SS}(S_c, S_{mc}) &= -2 \cdot \frac{n_\eta}{p_c} \cdot \left(\frac{S_{mc} - 1}{\frac{S_{mc}}{p_{mc}} - S_{mc} + 1} + 1\right) \cdot \frac{\frac{1-p_c}{p_c}}{\left(\frac{S_c}{p} - S_c + 1\right)^3} \\ &= -2 \cdot n_\eta \cdot \frac{\tilde{p}_c^{*3}}{p_c} \cdot \left(\tilde{p}_{mc}^*(S_{mc} - 1) + 1\right) \cdot \frac{1 - p_c}{p_c}, \\ g_{S_{mc}S_{mc}}(S_c, S_{mc}) &= -2 \cdot \frac{n_\eta}{p_{mc}} \cdot \left(\frac{S_c - 1}{\frac{S_c}{p_c} - S_c + 1} + 1\right) \cdot \frac{\frac{1-p_{mc}}{p_{mc}}}{\left(\frac{S_{mc}}{p_{mc}} - S_{mc} + 1\right)^3} \\ &= -2 \cdot n_\eta \cdot \frac{\tilde{p}_{mc}^{*\,3}}{p_{mc}} \cdot \left(\tilde{p}_c^*(S_c - 1) + 1\right) \cdot \frac{1 - p_{mc}}{p_{mc}}. \end{aligned}$$

After several reformulations, this leads to

$$\begin{aligned} E(n_{total}) &= E[g(S_c, S_{mc})] \\ &= n_\eta \cdot \left(\tilde{p}_c(\mu_c - 1) + 1\right) \cdot \left(\tilde{p}_{mc}(\mu_{mc} - 1) + 1\right) \\ &\quad \cdot \left(1 - \frac{1 - p_c}{p_c} \cdot \tilde{p}_c^2 - \frac{1 - p_{mc}}{p_{mc}} \cdot \tilde{p}_{mc}^2\right). \end{aligned}$$

Solving for n_η then gives the result. Note that the approximation in the sense of the second order Taylor expansion here comes in addition to potential inaccuracies already present in the sampling based on fixed cluster sizes.

□

Remark B.1 (**Expectation of distance between two random points inside the unit circle**). *For points uniformly distributed inside the unit circle, the expectation of the distance R between two randomly selected points is $\frac{128}{45\pi}$.*

Proof. Let h be the density for the vector difference R between the positions of two randomly selected points. It is possible to explicitly write it down: as may be derived from results given in Stoyan (1992) in the context of Matérn cluster processes, or as explained in Brown (2013) based on geometric considerations, it is

$$h(r) = \frac{r}{\pi}\left[4 \cdot \arctan\left(\frac{\sqrt{4-r^2}}{r}\right) - r \cdot \sqrt{4-r^2}\right].$$

After substitution $r = 2\cos(\frac{\vartheta}{2})$, this transforms to the much simpler

$$h(\vartheta) = \frac{2}{\pi} \cdot \sin(\vartheta)\left[\vartheta - \sin(\vartheta)\right].$$

Then, $E(R)$ can be calculated as

$$\int_0^2 r \cdot h(r)dr = \frac{2}{\pi}\int_0^\pi 2\cos\left(\frac{\vartheta}{2}\right) \cdot \sin(\vartheta)\left[\vartheta - \sin(\vartheta)\right] d\vartheta = \frac{128}{45\pi}.$$

□

Appendix C

Software

In the following, implementations of the methods developed in this thesis are documented.

For the spatial cluster analysis of Ras proteins, first in Section C.1 `R` functions are documented to sample point patterns from the double Matérn cluster process described in Section 4.2. Further, in Section C.3 MATLAB routines implementing the GAMMICS method described in Section 4.5 are documented.

For the integrative analysis of omics data, first in Section C.4, WinBUGS code may be found that implements the normal mixture model described in Section 5.3. Finally, Section C.5 contains documentation from the `R` package `epigenomix` available from Bioconductor, implementing methods to run the mixture model with normal and exponential (or gamma) components described in Section 5.4.

C.1 R Code for Point Process Simulations

Here, `R` functions are documented that permit to sample from the double Matérn cluster process. The functions `rPoissonCluster_double` and `rDoubleMatern` are analoga to the functions `rPoissonCluster` and `rNeymanScott` from the `spatstat` package implementing the Neyman-Scott cluster process, respectively, and use some of its functionality.

C.1.1 Documentation

`rDoubleMatern` *Simulate Double Matérn Cluster Process*

Description

Generate a random point pattern, a realisation of the Double Matérn cluster process.

Usage

```
rDoubleMatern(proportion_clustered,proportion_within_metaclusters,
    size,size_meta,min_size,min_size_meta,radius,radius_meta,
    size_type="random",point_density,
    transform_range_x=c(0,1), transform_range_y=c(0,1),
    plotting=TRUE,eps=10^(-14),detection_error_sd)
```

Arguments

`proportion_clustered`	Overall proportion of points lying in clusters
`proportion_within_metaclusters`	Overall proportion of points lying in meta-clusters (in and outside clusters)
`size`	Mean size of clusters
`size_meta`	Mean size of meta-clusters (a cluster counts as one point, i.e., size refers to parent level of clusters)
`min_size`	Minimum size of clusters
`min_size_meta`	Minimum size of meta-clusters
`radius`	Mean radius of clusters
`radius_meta`	Mean radius of meta-clusters
`size_type`	Either `fixed` or `random`: are the numbers of points per cluster and the cluster radii specified as fixed numbers or as random with a given mean?
`point_density`	Point density in points/μm^2

`transform_range_x`	Within which range should the resulting x coordinates lie? Should have equal width as `transform_range_y`
`transform_range_y`	Within which range should the resulting y coordinates lie? Should have equal width as `transform_range_x`
`plotting`	Should the simulated point pattern be plotted? `TRUE` or `FALSE`. legend: black=all (raw) points, blue=clustered points, red=points with error, green=clustered points with error
`eps`	Technical: maximum distance of offspring points from the location of the parent
`detection_error_sd`	Standard deviation of Gaussian distribution modeling detection error

Details

This algorithm generates a realisation of the Double Matérn cluster process, which as opposed to the Neyman-Scott cluster process allows for single points that are not in clusters as well as metaclusters, i.e. clusters that in turn contain clusters. The cluster mechanism is fixed to uniformly distribution of points within a cluster. The number of points per cluster/metacluster and the cluster/metacluster radii may be specified either as fixed or as random with a given mean, where points per cluster are following a poisson distribution and cluster radii a gamma distribution. For the case of random cluster/metacluster sizes, it is possible to specify a minimum size.

In absence of metaclusters, the algorithm generates first generation points with a Poisson point process. In presence of metaclusters, the algorithm generates immediately second generation points with a modified Neyman-Scott process. Then, a certain proportion of then existent points is replaced by random clusters of points. All points are combined together to yield a single point pattern which is then returned as the result of `rDoubleMatern`.

In case of random cluster sizes, the number of first generation points to sample is calculated from the input parameters based on a Taylor approximation.

If required, the intermediate stages of the simulation (the coordinates of first and second generation points and the allocation of third generation points to them) can also be extracted from the `raw` element of the `output` list. The `attr_parents_first_level` subelement gives the allocation of second generation points to first generation points, the `attr_parents_second_level` subelement gives the allocation of third generation points to second generation points.

Value

A list object with three elements: `input`, storing the specified input parameters (and calculated values resulting from the Taylor approximations); `output`, storing the point pattern with or without error along with its characteristics such as proportion of clustered points, mean cluster size and radius.

Author(s)

Martin Schaefer (martin.schaefer@udo.edu)

See Also

`rPoissonCluster_double`

Examples

```
proportion_clustered <- 0.7
proportion_within_metaclusters <- 0.9
size_meta <- 5
size <- 8
min_size_meta <- 2
min_size <- 3
size_type <- "random"
radius <- 15
radius_meta <- 80
transform_range_x=c(0,4000)
transform_range_y=c(0,4000)
point_density <- 125
plotting <-TRUE
eps <- 10^(-14)
```

```
detection_error_sd <- 20
simulation_double <- rDoubleMatern(proportion_clustered,
    proportion_within_metaclusters,size,size_meta,
    min_size,min_size_meta,radius,radius_meta,
    size_type,point_density,transform_range_x,
    transform_range_y, plotting=TRUE,eps=10^(-14),
    detection_error_sd)
```

`rPoissonCluster_double` *Simulate Poisson clusters on two levels*

Description

Generate a random point pattern with Poisson clusters on two levels.

Usage

```
rPoissonCluster_double(kappa, rmax, clust_meta, clust, radius_meta,
    radius, metaclusters, win = owin(c(0, 1), c(0, 1)), ...)
```

Arguments

`kappa`	Intensity of the Poisson process of cluster centers. A single positive number, a function, or a pixel image.
`rmax`	Maximum radius of a random cluster
`clust_meta`	A function which generates random meta-clusters
`clust`	A function which generates random clusters
`radius_meta`	Fixed or mean meta-cluster radius
`radius`	Fixed or mean cluster radius
`metaclusters`	Should meta-clusters be simulated? (either `TRUE` or `FALSE`)
`win`	Window in which to simulate the pattern. An object of class `owin` or something acceptable to `as.owin`
`...`	Arguments passed to the `rcluster` argument of `rNeymanScott`, which the function calls.

Details

This function acts mainly as auxiliary function for `rDoubleMatern`, it is the function which actually samples the point pattern.

Value

The simulated point pattern. For further details on the output format cf. `rPoissonCluster`.

Author(s)

Martin Schaefer (martin.schaefer@udo.edu)

See Also

`rDoubleMatern_double`

C.1.2 Code

Function rDoubleMatern

```
rDoubleMatern <- function(proportion_clustered,proportion_within_metaclusters,
    size,size_meta,min_size,min_size_meta,radius,radius_meta,size_type="random",
    point_density,transform_range_x=c(0,1), transform_range_y=c(0,1),
    plotting=TRUE,eps=10^(-14),detection_error_sd){
require(spatstat)
require(mvtnorm)
transform_range_x_diff <- transform_range_x[2] - transform_range_x[1]
transform_range_y_diff <- transform_range_y[2] - transform_range_y[1]
nr_points <- point_density*transform_range_x_diff/1000*
    transform_range_y_diff/1000
transform_range_x <- c(transform_range_x[1]+radius_meta,
    transform_range_x[2]-radius_meta)
transform_range_y <- c(transform_range_y[1]+radius_meta,
    transform_range_y[2]-radius_meta)
transform_range_x_diff <- transform_range_x[2] - transform_range_x[1]
transform_range_y_diff <- transform_range_y[2] - transform_range_y[1]
if(transform_range_x_diff != transform_range_y_diff){stop("Provided
    image region is not quadratic!")}
size_minus <- size - min_size
size_minus_meta <- size_meta - min_size_meta
if(proportion_within_metaclusters == 0){
    anteil_meta <- 0
} else {
    anteil_meta <- 1/(size_meta/proportion_within_metaclusters-size_meta+1)
}
if(proportion_clustered == 0){
    anteil <- 0
} else {
    anteil <- 1/(size/proportion_clustered-size+1)
}
status_meta <- status <- parent <- numeric()
nummer <- 0
```

```
if(size_type=="fixed"){
clust <- function(x0, y0, radius) {
    u <- runif(1)
    if(u <= anteil){
         status <<- c(status,1)
         return(runifdisc(size_minus + min_size,
            radius=radius/transform_range_x_diff,centre=c(x0, y0))) }
    else {
        status <<- c(status,0)
        object <- runifdisc(1, eps, radius=radius, centre=c(x0, y0))
        object$x <- x0
        object$y <- y0
        return(object)
    }
}
clust_meta <- function(x0, y0, radius) {
    u <- runif(1)
    if(u <= anteil_meta){
         status_meta <<- c(status_meta,1)
         return(runifdisc(size_minus_meta + min_size_meta,
            radius=radius/transform_range_x_diff, centre=c(x0, y0))) }
    else {
        status_meta <<- c(status_meta,0)
        object <- runifdisc(1, eps, radius=radius, centre=c(x0, y0))
        object$x <- x0
        object$y <- y0
        return(object)
    }
}
}
if(size_type=="random"){
clust <- function(x0, y0, radius) {
    u <- runif(1)
    if(u <= anteil){
         status <<- c(status,1)
         return(runifdisc(rpois(n=1,lambda=size_minus) + min_size,
```

```
            radius=rgamma(n=1,shape=radius,scale=1)/transform_range_x_diff,
            centre=c(x0, y0))) }
    else {
        status <<- c(status,0)
        object <- runifdisc(1, eps, radius=radius, centre=c(x0, y0))
        object$x <- x0
        object$y <- y0
        return(object)
    }
}
clust_meta <- function(x0, y0, radius) {
    u <- runif(1)
    if(u <= anteil_meta){
         status_meta <<- c(status_meta,1)
         return(runifdisc(rpois(n=1,lambda=size_minus_meta) + min_size_meta,
         radius=rgamma(n=1,shape=radius,scale=1)/transform_range_x_diff,
         centre=c(x0, y0))) }
    else {
        status_meta <<- c(status_meta,0)
        object <- runifdisc(1, eps, radius=radius, centre=c(x0, y0))
        object$x <- x0
        object$y <- y0
        return(object)
    }
}
}
if(proportion_within_metaclusters == 0){
    if(proportion_clustered == 0){
        aprox_number <- nr_points
    } else {
        aprox_number <- nr_points/ ((anteil*(size-1)+1) *
        (1 - (1/proportion_clustered-1)*anteil^2 ))
    }
    metaclusters <- FALSE
} else {
    if(proportion_clustered == 0){
```

```
        aprox_number <- nr_points/ ((anteil_meta*(size_meta-1)+1) *
            (1 - (1-proportion_within_metaclusters)/
            proportion_within_metaclusters*anteil_meta^2))
    } else {
        aprox_number <- nr_points/ (((anteil*(size-1)+1)*
        (anteil_meta*(size_meta-1)+1)) * (1 - (1/proportion_clustered-1)*
        anteil^2 - ((1-proportion_within_metaclusters)/
        proportion_within_metaclusters)*anteil_meta^2))
    }
    metaclusters <- TRUE
}
prozess <- rPoissonCluster_double(kappa=aprox_number,
rmax=radius_meta/transform_range_x_diff, clust_meta=clust_meta, clust=clust,
radius_meta=radius_meta, radius=radius, metaclusters=metaclusters,
win=owin(xrange=c(0,1),yrange=c(0,1)))
#############################################################################
welche <- duplicated(attr(prozess, "parentid"))
ind <- which(welche ==TRUE)
welche[ind-1] <- TRUE
geclustert <- which(welche == TRUE)
monomere <- which(welche == FALSE)
wo_cluster <- which(table(attr(prozess, "parentid")) > 1)
nr_clust <- length(wo_cluster)
faktor_x <- transform_range_x_diff
faktor_y <- transform_range_y_diff
x <- prozess$x*faktor_x+transform_range_x[1]#
y <- prozess$y*faktor_y+transform_range_y[1]#
if(proportion_clustered == 0){
    mean_radius <- median_radius <- mean_cluster_size <- median_cluster_sizeNA
} else {
    radien <- numeric(nr_clust)
    for(i in 1:nr_clust){
        welche_drin <- which(attr(prozess, "parentid") ==
            as.numeric(names(wo_cluster))[i])
        distanzen <- pairdist(cbind(x[welche_drin],y[welche_drin]))
        radien[i] <- max(distanzen)/2
```

```
    }
    mean_radius <- mean(radien)
    median_radius <- median(radien)
    mean_cluster_size <- mean(table(attr(prozess,"parentid"))[wo_cluster])
    median_cluster_size <- median(table(attr(prozess,"parentid"))[wo_cluster])
}
#############################################################################
welche <- duplicated(attr(attr(prozess, "parents"),"parentid"))
ind <- which(welche ==TRUE)
welche[ind-1] <- TRUE
gemetaclustert <- which(welche == TRUE)
metamonomere <- which(welche == FALSE)
wo_metacluster <- which(table(attr(attr(prozess, "parents"),"parentid")) > 1)
nr_metaclust <- length(wo_metacluster)
faktor_x <- transform_range_x_diff
faktor_y <- transform_range_y_diff
x_meta <- attr(prozess, "parents")$x*faktor_x+transform_range_x[1]#
y_meta <- attr(prozess, "parents")$y*faktor_y+transform_range_y[1]#
if(proportion_within_metaclusters == 0){
    mean_metaradius <- median_metaradius <- NA
    mean_metacluster_size <- median_metacluster_size <- NA
} else {
    metaradien <- numeric(nr_metaclust)
    for(i in 1:nr_metaclust){
        welche_drin <- which(attr(attr(prozess, "parents"),"parentid") ==
        as.numeric(names(wo_metacluster))[i])
        distanzen <- pairdist(cbind(x_meta[welche_drin],y_meta[welche_drin]))
        metaradien[i] <- max(distanzen)/2
    }
    mean_metaradius <- mean(metaradien)
    median_metaradius <- median(metaradien)
    mean_metacluster_size <- mean(table(attr(attr(prozess, "parents"),
        "parentid"))[wo_metacluster])
    median_metacluster_size <- median(table(attr(attr(prozess, "parents"),
    "parentid"))[wo_metacluster])
}
```

```
koordinaten <- matrix(numeric(2*prozess$n), ncol=2)
for(j in 1:prozess$n){
    koordinaten[j,] <- as.numeric(rmvnorm(1, mean = c(x[j],y[j]),
    sigma = diag(2)*detection_error_sd^2))
}
x_with_error <- koordinaten[,1]
y_with_error <- koordinaten[,2]
if(plotting== TRUE){
    par(mar=c(4, 4, 2, 2) + 0.1)
    plot(x,y, xlim=transform_range_x, xlab="", ylab="",
    ylim=transform_range_y, cex.axis=1.3, cex.lab=1.3, pch=20)
    par(new=TRUE)
    plot(x[geclustert],y[geclustert],xlim=transform_range_x,
    ylim=transform_range_y, xlab="x coordinate (nm)",
    ylab="y coordinate (nm)", col="blue", cex.axis=1.3, cex.lab=1.3, pch=20)
    par(new=TRUE)
    plot(x_with_error,y_with_error,xlim=transform_range_x,
    ylim=transform_range_y, xlab="x coordinate (nm)",
    ylab="y coordinate (nm)", col="red", cex.axis=1.3, cex.lab=1.3, pch=20)
    par(new=TRUE)
    plot(x_with_error[geclustert],y_with_error[geclustert],
    xlim=transform_range_x, ylim=transform_range_y, xlab="x coordinate (nm)",
    ylab="y coordinate (nm)", col="green", cex.axis=1.3, cex.lab=1.3, pch=20)
}
input <- list(proportion_clustered=proportion_clustered,
proportion_within_metaclusters=proportion_within_metaclusters,
mean_cluster_size_meta=size_meta,mean_cluster_size=size,
mean_cluster_radius_meta=radius_meta,mean_cluster_radius=radius,
size_type=size_type,nr_points=nr_points,point_density=point_density,
detection_error_sd=detection_error_sd,
transform_range_x=transform_range_x,transform_range_y=transform_range_y,
eps=eps,nr_first_parents=aprox_number,
proportion_replaced_grandparents=anteil_meta,
proportion_replaced_parents=anteil,
min_size=min_size,min_size_meta=min_size_meta)
```

```
output_all <- list(x = x, y= y, x_with_error = x_with_error,
y_with_error = y_with_error, x_clust = x[geclustert], y_clust = y[geclustert],
x_clust_with_error = x_with_error[geclustert],
y_clust_with_error = y_with_error[geclustert],
clust = geclustert, nr_clust = nr_clust,
nr_metaclust = nr_metaclust, proportion_clustered=length(geclustert)/length(x),
cluster_size=list(mean=mean_cluster_size, median=median_cluster_size),
metacluster_size=list(mean=mean_metacluster_size,
median=median_metacluster_size),
cluster_radius=list(mean=mean_radius, median=median_radius),
metacluster_radius=list(mean=mean_metaradius, median=median_metaradius),
raw=list(x_first_level_parents=attr(attr(prozess, "parents"), "parents")$x,
y_first_level_parents=attr(attr(prozess, "parents"), "parents")$y,
x_second_level_parents=attr(prozess, "parents")$x,
y_second_level_parents=attr(prozess, "parents")$y,
attr_parents_first_level=attr(attr(prozess, "parents"),
"parentid"),attr_parents_second_level=attr(prozess, "parentid"))
)
return(list(input=input, output=output_all))
}
```

Function `rPoissonCluster_double`

```
rPoissonCluster_double <- function(kappa, rmax, clust_meta, clust, radius_meta,
radius, metaclusters, win = owin(c(0, 1), c(0, 1)),..., lmax = NULL)
{
    win <- as.owin(win)
    frame <- boundingbox(win) #bounding.box(win)
    dilated <- owin(frame$xrange + c(-rmax, rmax), frame$yrange +
        c(-rmax, rmax))
    if (is.im(kappa) && !is.subset.owin(dilated, as.owin(kappa)))
        stop(paste("The window in which the image", sQuote("kappa"),
            "is defined\n", "is not large enough to contain the dilation
            of the window", sQuote("win")))
    if(metaclusters){
        parents <- rNeymanScott(kappa=kappa, expand=1, rcluster=clust_meta,
```

```
        radius=radius_meta, win = dilated, lmax = NULL)
    } else {
        parents <- rpoispp(lambda=kappa, lmax = lmax, win = dilated)
    }
    result <- NULL
    np <- parents$n
    if (np > 0) {
        xparent <- parents$x
        yparent <- parents$y
        for (i in seq_len(np)) {
            cluster <- clust(xparent[i], yparent[i], radius=radius)
            if (!inherits(cluster, "ppp"))
                cluster <- ppp(cluster$x, cluster$y, window = frame,
                  check = FALSE)
            if (cluster$n > 0) {
                cluster <- cluster[win]
                if (is.null(result)) {
                  result <- cluster
                  parentid <- rep(1, cluster$n)
                }
                else {
                  result <- superimpose(result, cluster, W = win)
                  parentid <- c(parentid, rep(i, cluster$n))
                }
            }
        }
    }
    else {
        result <- ppp(numeric(0), numeric(0), window = win)
        parentid <- integer(0)
    }
    attr(result, "parents") <- parents
    attr(result, "parentid") <- parentid
    return(result)
}
```

C.2 R Code for Monte Carlo Tests

Here, `R` functions are documented to perform Monte Carlo tests for extreme cases regarding p_c using either the H-function or paircorrelation function. The functions `MC_test_for_no_clustering` and `MC_test_for_only_clustering` permit to test for the cases $p_c = 0$ and $p_c = 1$, respectively. They make use of functionality from the `spatstat` package.

C.2.1 Documentation

`MC_test_no_clustering` *Perform a Monte Carlo test for no clustering*

Description

Perform a Monte Carlo test for the case $p_c = 0$.

Usage

```
MC_test_no_clustering(X,Y,withError,confidence,method,
    iterations,relevant_radius,range_x=NULL,range_y=NULL)
```

Arguments

`X`	X-coordinates of the data points
`Y`	Y-coordinates of the data points
`withError`	Are point patterns to be simulated a detection error? (yes=1, no=0)
`confidence`	The confidence level of the confidence interval
`method`	Should the test be made based on the H-function (`method="Hfun"`) or the pair correlation function (`method="pcfun"`)?
`iterations`	Number of iterations
`relevant_radius`	Maximum radius up to which it will be checked numerically whether the function based on the data lies within the confidence interval or not

`range_x` The x range for the simulated images. If not specified, the function makes an "intelligent guess" based on the ranges of the protein values.

`range_y` The y range for the simulated images. If not specified, the function makes an "intelligent guess" based on the ranges of the protein values.

Details

This algorithm performs a Monte Carlo test for the case of no clustering, i.e. $p_c = 0$. To calculate the confidence interval, point patterns without clustering are simulated using the function `rpoispp`, and relevant parameters of this function can be specified.

Value

A plot with the H-function or pair correlation function of the data and the confidence interval for the case of no clustering. The result of the test is printed on the screen.

Author(s)

Martin Schaefer (martin.schaefer@udo.edu)

See Also

`rpoispp`

Examples

```
simulation_double <- rDoubleMatern(proportion_clustered=0.7,
    proportion_within_metaclusters=0.9,size=8,size_meta=5,
    min_size=2,min_size_meta=2,radius=15,radius_meta=80,
    size_type="random",point_density=125,transform_range_x=c(0,4000),
    transform_range_y=c(0,4000), plotting=TRUE,eps=10^(-14),
    detection_error_sd=20)
withError <- 1
confidence <- 0.99
method <- "Hfun"
iterations <- 500
relevant_radius <- 100
range_x <- c(0,4000)
range_y <- c(0,4000)
```

```
MC_test_no_clustering(X=simulation_double$output$x_with_error,
    Y=simulation_double$output$x_with_error,withError,confidence,
    method,iterations,relevant_radius)
```

`MC_test_only_clustering` *Perform a Monte Carlo test for 100% clustering*

Description

Perform a Monte Carlo test for the case $p_c = 1$.

Usage

```
MC_test_only_clustering(X,Y,withError,confidence,size_type,method,
    iterations,relevant_radius,pwm,sm,rm,pd,range_x=NULL,
    range_y=NULL,size_estimate100,radius_estimate100)
```

Arguments

`X`	X-coordinates of the data points
`Y`	Y-coordinates of the data points
`withError`	Are point patterns to be simulated a detection error? (yes=1, no=0)
`confidence`	The confidence level of the confidence interval
`size_type`	Is the cluster radius fix (`size_type="fixed"`) or to be drawn from a Poisson distribution (`size_type="random"`) during the test?
`method`	Should the test be made based on the H-function (`method="Hfun"`) or the pair correlation function (`method="pcfun"`)?
`iterations`	Number of iterations
`relevant_radius`	Maximum radius up to which it will be checked numerically whether the function based on the data lies within the confidence interval or not
`pwm`	proportion of proteins in metaclusters
`sm`	size of metaclusters

`rm`	radius of metaclusters
`pd`	point density
`range_x`	The x range for the simulated images. If not specified, the function makes an "intelligent guess" based on the ranges of the protein values.
`range_y`	The y range for the simulated images. If not specified, the function makes an "intelligent guess" based on the ranges of the protein values.
`size_estimate100`	Estimate of the mean cluster size in the data conditional on $p_c = 1$
`radius_estimate100`	Estimate of the mean cluster radius in the data conditional on $p_c = 1$

Details

This algorithm perform a Monte Carlo test for the case of no clustering, i.e. $p_c = 1$. To calculate the confidence interval, point patterns with clustering are simulated using the function `rDoubleMatern`, and relevant parameters of this function can be specified. For the clusters in the point patterns simulated during the test, estimates of mean cluster size and radius have to be specified. They may be obtained, e.g., by running DBSCAN from the R package `fpc` on the data, setting $MinPts = 2$ and ε equal to the maximum distance of any point to its nearest neighbour. The proportion, size and radius of metaclusters also have to be specified. Since no means exist to estimate them, it is recommended to perform the test for a representative grid of relevant values.

Value

A plot with the H-function or pair correlation function of the data and the confidence interval for the case of no clustering. The result of the test is printed on the screen.

Author(s)

Martin Schaefer (martin.schaefer@udo.edu)

See Also

`rDoubleMatern`

Examples

```
simulation_double <- rDoubleMatern(proportion_clustered=0.7,
    proportion_within_metaclusters=0.9,size=8,size_meta=5,
    min_size=2,min_size_meta=2,radius=15,radius_meta=80,
    size_type="random",point_density=125,transform_range_x=c(0,4000),
    transform_range_y=c(0,4000), plotting=TRUE,eps=10^(-14),
    detection_error_sd=20)
withError <- 1
confidence <- 0.99
size_type <- "random"
method <- "Hfun"
iterations <- 500
relevant_radius <- 100
range_x <- c(0,4000)
range_y <- c(0,4000)
pwm <- 0.5
sm <- 20
rm <- 100
pd <- length(simulation_double$output$x_with_error)/
    ((range_x[2]-range_x[1])*(range_y[2]-range_y[1])/1e+06)
size_estimate100 <- 20
radius_estimate100 <- 100
MC_test_only_clustering(X=simulation_double$output$x_with_error,
    Y=simulation_double$output$x_with_error,withError,confidence,
    size_type,method,iterations,relevant_radius,pwm,sm,rm,pd,
    range_x=NULL,range_y=NULL,size_estimate100,radius_estimate100)
```

C.2.2 Code

Function `MC_test_no_clustering`

```
MC_test_no_clustering <- function(X,Y,withError,confidence,method,iterations,
    relevant_radius,range_x=NULL,range_y=NULL){
par(mar=c(5,6,4,2)+0.1)
library(spatstat)
#################### Calculation of H-Function for Data ####################
range_x <- c(floor(range(X)[1]),ceiling(range(X)[2]))
range_y <- c(floor(range(Y)[1]),ceiling(range(Y)[2]))
range_x_diff <- range_x[2] - range_x[1]
range_y_diff <- range_y[2] - range_y[1]
if(range_x_diff > range_y_diff){
    range_y[2] <- range_y[1] + range_x_diff
} else {
    range_x[2] <- range_x[1] + range_y_diff
}
win <- owin(range_x,range_y)
daten <- ppp(x=X,y=Y,window=win)
K_daten <- Kinhom(daten, nlarge=1000)
pcf_daten <- pcf(K_daten)$pcf
################## Calculation of H-Function for Simulations ##################
bw.diggle_value <- bw.diggle(daten)
dens <- density(daten)#,sigma=bw.diggle_value)
sim1 <- rpoispp(lambda=dens)
sim_dings <- rpoispp(lambda=daten$n/((daten$window$xrange[2]-
    daten$window$xrange[1])*(daten$window$yrange[2]-daten$window$yrange[1])),
    win=win)
sim1$n <- sim_dings$n
sim1$x <- sim_dings$x
sim1$y <- sim_dings$y
K_A1 <- Kinhom(sim1, nlarge=1000)
if(method == "pcfun"){
  erg1 <- pcf(K_A1)$pcf
} else {
```

```
  erg1 <- K_A1$border
}
sim <- list()
erg <- matrix(numeric(iterations*length(erg1)), nrow=iterations)
ergebnisse <- list()
ind <- 1
for(i in 1:iterations){
  if((i %% 50) == 0){print(i)}
  sim_temp <- rpoispp(lambda=daten$n/((daten$window$xrange[2]-
    daten$window$xrange[1])*(daten$window$yrange[2]-daten$window$yrange[1])),
    win=win)
  sim1$n <- sim_temp$n
  sim1$x <- sim_temp$x
  sim1$y <- sim_temp$y
  K_A <- Kinhom(sim1)
  if(method == "pcfun"){
    erg[i,] <- pcf(K_A)$pcf
  } else {
    erg[i,] <- K_A$border
  }
}
gib_grenzen <- function(x){
    quantile(x, probs=c((1-confidence)/2,1-(1-confidence)/2))
}
KI_Werte <- apply(erg,2,gib_grenzen)
############ Plot results ###############
par(mar=c(5,6,4,2)+0.1)
if(method == "pcfun"){
  maxwert_pcf <- max(max(KI_Werte[2,]),max(pcf_daten))
  minwert_pcf <- min(min(KI_Werte[1,]),min(pcf_daten))
  grenze_inds <- which(pcf(K_daten)$r <= relevant_radius)
  plot(pcf(K_daten)$r, pcf_daten, xlab="r", ylab="pair correlation function",
  main="", xlim=c(0,max(pcf(K_daten)$r)), ylim=c(minwert_pcf,maxwert_pcf),
  lwd=2, cex.axis=1.4, cex.lab=1.4, type="l")
  par(new=TRUE)
  plot(pcf(K_A1)$r, KI_Werte[1,], xlim=c(0,max(pcf(K_A1)$r)),
```

```
  ylim=c(minwert_pcf,maxwert_pcf), type="l", xlab="", ylab="", lty=2, lwd=2,
  cex.axis=1.4, cex.lab=1.4, col="red")
  par(new=TRUE)
  plot(pcf(K_A1)$r, KI_Werte[2,], xlim=c(0,max(pcf(K_A1)$r)),
  ylim=c(minwert_pcf,maxwert_pcf), type="l", xlab="", ylab="", lty=2, lwd=2,
  cex.axis=1.4, cex.lab=1.4, col="red")
  abline(h=1, lty=3, lwd=2)
  if(any((pcf_daten[grenze_inds] - KI_Werte[1,grenze_inds]) < 0) |
    any((KI_Werte[2,grenze_inds] - pcf_daten[grenze_inds]) < 0)){
    print(paste("The data do not lie within the MC confidence interval up to
    a relevant radius of", relevant_radius, "nm. It can be assumed that
    clustering is present."))
  } else {
    print(paste("The data lie within the MC confidence interval up to a
    relevant radius of", relevant_radius, "nm. It can be assumed that no
    clustering is present."))
  }
} else {
  welche_border <- which(K_daten$border < 0)
  welche_werte1 <- which(KI_Werte[1,] < 0)
  welche_werte2 <- which(KI_Werte[2,] < 0)
  K_daten$border[welche_border] <- 0
  KI_Werte[1,][welche_werte1] <- 0
  KI_Werte[2,][welche_werte2] <- 0
  maxwert_K <- max(max(sqrt(KI_Werte[2,]/pi) - K_A$r),
    max(sqrt(K_daten$border/pi) - K_daten$r))
  minwert_K <- min(min(sqrt(KI_Werte[1,]/pi) - K_A$r),
    min(sqrt(K_daten$border/pi) - K_daten$r))
  grenze_inds <- which(K_daten$r <= relevant_radius)
  plot(K_daten$r, sqrt(K_daten$border/pi) - K_daten$r, xlab="r",
    ylab=expression(hat(H)(r)==sqrt((hat(K)(r)/pi))-r), main="",
    xlim=c(0,max(K_daten$r)), ylim=c(minwert_K,maxwert_K), lwd=2,
    cex.axis=1.4, cex.lab=1.4, type="l")
  par(new=TRUE)
  plot(K_A$r, sqrt(KI_Werte[1,]/pi) - K_A$r, xlim=c(0,max(K_daten$r)),
  ylim=c(minwert_K,maxwert_K), type="l", xlab="", ylab="", lty=2, lwd=2,
```

```
  cex.axis=1.4, cex.lab=1.4, col="red")
  par(new=TRUE)
  plot(K_A$r, sqrt(KI_Werte[2,]/pi) - K_A$r, xlim=c(0,max(K_daten$r)),
  ylim=c(minwert_K,maxwert_K), type="l", xlab="", ylab="", lty=2, lwd=2,
  cex.axis=1.4, cex.lab=1.4, col="red")
  abline(h=0, lty=3, lwd=2)
  if(any((K_daten$border[grenze_inds] - KI_Werte[1,grenze_inds]) < 0) |
    any((KI_Werte[2,grenze_inds] - K_daten$border[grenze_inds]) < 0)){
    print(paste("The data do not lie within the MC confidence interval up to
    a relevant radius of", relevant_radius, "nm. It can be assumed that
    clustering is present."))
  } else {
    print(paste("The data lie within the MC confidence interval up to a
    relevant radius of", relevant_radius, "nm. It can be assumed that
    no clustering is present."))
  }
}
}
```

Function `MC_test_only_clustering`

```
MC_test_only_clustering <- function(X,Y,withError,confidence,size_type,method,
    iterations,relevant_radius,pwm,sm,rm,pd,range_x=NULL,range_y=NULL,
    size_estimate100,radius_estimate100){
library(spatstat)
library(mvtnorm)
################## Calculation of H-Function for Data ##################
    range_x <- c(floor(range(X)[1]),ceiling(range(X)[2]))
    range_y <- c(floor(range(Y)[1]),ceiling(range(Y)[2]))
    range_x_diff <- range_x[2] - range_x[1]
    range_y_diff <- range_y[2] - range_y[1]
    if(range_x_diff > range_y_diff){
        range_y[2] <- range_y[1] + range_x_diff
    } else {
        range_x[2] <- range_x[1] + range_y_diff
    }
```

```
    win <- owin(range_x,range_y)
    daten <- ppp(x=X,y=Y,window=win)
K_daten <- Kinhom(daten, nlarge=1000)
pcf_daten <- pcf(K_daten)$pcf
######## Calculation of H-Function for Data Simulations ###############
bw.diggle_value <- bw.diggle(daten)
dens <- density(daten)
  nr_points <- length(X)
sim0 <- rpoispp(lambda=nr_points)
sim_dings <- rDoubleMatern(proportion_clustered=1,
    proportion_within_metaclusters=pwm,size=size_estimate100,size_meta=sm,
    min_size=3,min_size_meta=2,radius=radius_estimate100,radius_meta=rm,
    size_type=size_type,point_density=pd,transform_range_x=
    round(c(range_x[1]+radius_estimate100*sqrt(2),range_x[2]-
        radius_estimate100*sqrt(2))), transform_range_y=
        round(c(range_y[1]+radius_estimate100*sqrt(2),
        range_y[2]-radius_estimate100*sqrt(2))), plotting=FALSE,
        eps=10^(-14),detection_error_sd=20)# )#
sim0$n <- length(sim_dings$output$x)
sim0$window$xrange <- c(range_x[1]+radius_estimate100*sqrt(2),
    range_x[2]-radius_estimate100*sqrt(2))
sim0$window$yrange <- c(range_y[1]+radius_estimate100*sqrt(2),
    range_y[2]-radius_estimate100*sqrt(2))
if(withError){
  sim0$x <- sim_dings$output$x_with_error
  sim0$y <- sim_dings$output$y_with_error
} else {
  sim0$x <- sim_dings$output$x
  sim0$y <- sim_dings$output$y
}
K_A1 <- Kinhom(sim0, nlarge=1000)
if(method == "pcfun"){
  erg1 <- pcf(K_A1)$pcf
} else {
  erg1 <- K_A1$border
}
```

```
sim <- list()
erg <- matrix(numeric(iterations*length(erg1)), nrow=iterations)
ergebnisse <- list()
ind <- 1
for(i in 1:iterations){
  if((i %% 50) == 0){print(i)}
  sim_temp <- rDoubleMatern(proportion_clustered=1,
    proportion_within_metaclusters=pwm,size=size_estimate100,size_meta=sm,
    min_size=3,min_size_meta=2,radius=radius_estimate100,radius_meta=rm,
    size_type=size_type,point_density=pd,transform_range_x=
    round(c(range_x[1]+radius_estimate100*sqrt(2),range_x[2]-
    radius_estimate100*sqrt(2))), transform_range_y=
    round(c(range_y[1]+radius_estimate100*sqrt(2),
    range_y[2]-radius_estimate100*sqrt(2))), plotting=FALSE,eps=10^(-14),
    detection_error_sd=20)# )
  sim1 <- sim0
  sim1$n <- length(sim_temp$output$x)
  sim1$window$xrange <- c(range_x[1]+radius_estimate100*sqrt(2),
    range_x[2]-radius_estimate100*sqrt(2))
  sim1$window$yrange <- c(range_y[1]+radius_estimate100*sqrt(2),
    range_y[2]-radius_estimate100*sqrt(2))
  if(withError){
    sim1$x <- sim_temp$output$x_with_error
    sim1$y <- sim_temp$output$y_with_error
  } else {
    sim1$x <- sim_temp$output$x
    sim1$y <- sim_temp$output$y
  }
  K_A <- Kinhom(sim1, nlarge=1000)
  if(method == "pcfun"){
    erg[i,] <- pcf(K_A)$pcf
  } else {
    erg[i,] <- K_A$border
  }
}
gib_grenzen <- function(x){
```

```
    quantile(x, probs=c((1-confidence)/2,1-(1-confidence)/2))
}
KI_Werte <- apply(erg,2,gib_grenzen)
##################### Plot results #####################################
par(mar=c(5,6,4,2)+0.1)
if(method == "pcfun"){
  maxwert_pcf <- max(max(KI_Werte[2,]),max(pcf_daten))
  minwert_pcf <- min(min(KI_Werte[1,]),min(pcf_daten))
  maxwertx_pcf <- min(max(K_A$r),max(K_daten$r))
  grenze_inds <- which(pcf(K_daten)$r <= relevant_radius)
  plot(pcf(K_daten)$r, pcf_daten, xlab="r", ylab="pair correlation function",
    main="", xlim=c(0,maxwertx_pcf), ylim=c(minwert_pcf,maxwert_pcf), lwd=2,
    cex.axis=1.4, cex.lab=1.4, type="l")
  par(new=TRUE)
  plot(pcf(K_A)$r, KI_Werte[1,], xlim=c(0,maxwertx_pcf),
    ylim=c(minwert_pcf,maxwert_pcf), type="l", xlab="", ylab="", lty=2, lwd=2,
    cex.axis=1.4, cex.lab=1.4, col="red")
  par(new=TRUE)
  plot(pcf(K_A)$r, KI_Werte[2,], xlim=c(0,maxwertx_pcf),
    ylim=c(minwert_pcf,maxwert_pcf), type="l", xlab="", ylab="", lty=2, lwd=2,
    cex.axis=1.4, cex.lab=1.4, col="red")
  abline(h=1, lty=3, lwd=2)
  if(any((pcf_daten[grenze_inds] - KI_Werte[1,grenze_inds]) < 0) |
    any((KI_Werte[2,grenze_inds] - pcf_daten[grenze_inds]) < 0)){
    print(paste("The data do not lie within the MC confidence interval up to
    a relevant radius of", relevant_radius, "nm. It can be assumed that
    clustering is present."))
  } else {
    print(paste("The data lie within the MC confidence interval up to a
    relevant radius of", relevant_radius, "nm. It can be assumed that
    no clustering is present."))
  }
} else {
  welche_border <- which(K_daten$border < 0)
  welche_werte1 <- which(KI_Werte[1,] < 0)
  welche_werte2 <- which(KI_Werte[2,] < 0)
```

```
  K_daten$border[welche_border] <- 0
  KI_Werte[1,][welche_werte1] <- KI_Werte[2,][welche_werte2] <- 0
  maxwert_K <- max(max(sqrt(KI_Werte[2,]/pi) - K_A$r),
    max(sqrt(K_daten$border/pi) - K_daten$r))
  minwert_K <- min(min(sqrt(KI_Werte[1,]/pi) - K_A$r),
    min(sqrt(K_daten$border/pi) - K_daten$r))
  maxwertx_K <- min(max(K_daten$r),max(K_A$r))
  grenze_inds <- which(K_daten$r <= relevant_radius)
   plot(K_daten$r, sqrt(K_daten$border/pi) - K_daten$r, xlab="r",
    ylab=expression(hat(H)(r)==sqrt((hat(K)(r)/pi))-r), main="",
    xlim=c(0,maxwertx_K), ylim=c(minwert_K,maxwert_K), lwd=2,
    cex.axis=1.4, cex.lab=1.4, type="l")
  par(new=TRUE)
  plot(K_A$r, sqrt(KI_Werte[1,]/pi) - K_A$r, xlim=c(0,maxwertx_K),
    ylim=c(minwert_K,maxwert_K), type="l", xlab="", ylab="", lty=2,
    lwd=2, cex.axis=1.4, cex.lab=1.4, col="red")
  par(new=TRUE)
  plot(K_A$r, sqrt(KI_Werte[2,]/pi) - K_A$r, xlim=c(0,maxwertx_K),
    ylim=c(minwert_K,maxwert_K), type="l", xlab="", ylab="", lty=2,
    lwd=2, cex.axis=1.4, cex.lab=1.4, col="red")
  abline(h=0, lty=3, lwd=2)
  if(any((K_daten$border[grenze_inds] - KI_Werte[1,grenze_inds]) < 0) |
    any((KI_Werte[2,grenze_inds] - K_daten$border[grenze_inds]) < 0)){
    print(paste("The data do not lie within the MC confidence interval
    up to a relevant radius of", relevant_radius, "nm. It can be assumed
    that clustering is present."))
  } else {
    print(paste("The data lie within the MC confidence interval up to
    a relevant radius of", relevant_radius, "nm. It can be assumed that
    no clustering is present."))
  }
}
}
```

C.3 MATLAB Code for GAMMICS

In the following, code to run the GAMMICS method is documented. It can be run on MATLAB employing the Statistics toolbox. The main code to run GAMMICS is given in Section C.3.1, auxiliary functions in Section C.3.2. An exemplary pipeline to run the code is given in Section C.3.3.

C.3.1 Main Code

```
tic
if simulated == 1 && detection_error == 1
    x_withBorder = [X_nm_with_error, Y_nm_with_error];
else
    x_withBorder = [X_nm, Y_nm];
end
%%%%%%%%%%%%%%%%%%%%%%%%%% WITH BORDER %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Calc distance matrix
x = x_withBorder;
[n,k]=size(x);
abstand = squareform(pdist(x_withBorder));
abstand = abstand.^2; %Quadrieren
abstand_temp = abstand;
%%%%%%%%%%%%%%%%%%%%%%%%%% WITHOUT BORDER %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% correction for edge effects
InBorderX = find((x_withBorder(:,1) < left + borderWidth) |
    (x_withBorder(:,1) > right - borderWidth));
InBorderY = find((x_withBorder(:,2) < bottom + borderWidth) |
    (x_withBorder(:,2) > top - borderWidth));
InBorder = union(InBorderX,InBorderY);
NotInBorder = setdiff(1:length(x_withBorder),InBorder);
x_withoutBorder = x_withBorder(NotInBorder,:);
if simulated == 1
    clustNotInBorder = intersect(clust,NotInBorder);
end
[n_alles,pstar]=size(x_withoutBorder);
```

```
abstand_temp2 = abstand_temp(:,NotInBorder);
sort_abstand_temp2_stern = sort(abstand_temp2);
minabstand_temp2_stern = sort_abstand_temp2_stern(knn+1,:);
minabstand_temp2_stern1 = sort_abstand_temp2_stern(2,:);
std_abstand2 = scaling_distances*mad(minabstand_temp2_stern);
std_abstand2_1 = scaling_distances*mad(minabstand_temp2_stern1);
minabstand_temp2 = minabstand_temp2_stern/std_abstand2;
minabstand_temp2_1 = minabstand_temp2_stern1/std_abstand2_1;
%%%%%%%%%%%%%%%%%% WITHOUT BORDER AND OUTLIERS %%%%%%%%%%%%%%%%%%%%%%%%
if lower_quantile == 0
       ausreisser_def = quantile(minabstand_temp2,upper_quantile);
else
    ausreisser_def = quantile(minabstand_temp2,upper_quantile)-
        quantile(minabstand_temp2,lower_quantile);
end
keine_ausreisser = find(minabstand_temp2 < ausreisser_def);
x = x_withoutBorder(keine_ausreisser,:);
[n,k]=size(x);
% Initialise distance matrix and binary variable
abstand_temp3 = abstand_temp2(:,keine_ausreisser);
sort_abstand_temp3_stern = sort(abstand_temp3);
minabstand_temp3_stern = sort_abstand_temp3_stern(knn+1,:);
minabstand_temp3_stern1 = sort_abstand_temp3_stern(2,:);
std_abstand3 = scaling_distances*mad(minabstand_temp3_stern);
std_abstand3_1 = scaling_distances*mad(minabstand_temp3_stern1);
minabstand = minabstand_temp3_stern/std_abstand3;
minabstand_1 = minabstand_temp3_stern1/std_abstand3_1;
final_ind = NotInBorder(keine_ausreisser);
final_ind_complement = setdiff(1:length(abstand_temp3),final_ind);
abstand = abstand(final_ind,final_ind);
%########################## a priori cutoff ##############################
cut_folge1_null = 0:0.001:max(minabstand);
pdfwerte0_1_null = gampdf(cut_folge1_null,shape0(1),scale0(1));
pdfwerte1_1_null = gampdf(cut_folge1_null,shape0(2),scale0(2));
pdf_differenz_null = pdfwerte1_1_null - pdfwerte0_1_null;
modus_ind_null = find(pdfwerte1_1_null == max(pdfwerte1_1_null));
```

```
cut_folge2_null = cut_folge1_null(modus_ind_null):0.001:max(minabstand);
pdfwerte0_2_null = gampdf(cut_folge2_null,shape0(1),scale0(1));
pdfwerte1_2_null = gampdf(cut_folge2_null,shape0(2),scale0(2));
pdf_differenz_null = pdfwerte1_2_null - pdfwerte0_2_null;
unter_null_null = find(pdf_differenz_null < 0);
grenze_null = cut_folge2_null(unter_null_null(1));
ind_transfer_null = find(abs(minabstand - grenze_null) ==
    min(abs(minabstand - grenze_null)));
grenze0 = min(minabstand_1(ind_transfer_null));
%%%%%%%%%%%%%%%%%%%%% initialze storage objects %%%%%%%%%%%%%%%%%%%
S.prop_distr=[];
S.prop=[];
S.size_qx_mit =[]; S.size_qx_ohne =[];
S.size_mit=[]; S.size_ohne=[];
S.radius_qx_mit = []; S.radius_qx_ohne = [];
S.radius_mit=[]; S.radius_ohne=[];
S.radius1_mit=[]; S.radius1_ohne=[];
S.nakt=[];S.mcr=[];
S.nr_clusters_mit=[]; S.nr_clusters_ohne=[];
S.grenze_mit=[]; S.grenze_ohne=[];
S.shape_nonclust=[];S.shape_clust=[];S.scale_nonclust=[];S.scale_clust=[];
S.shape_nonclust_accept=[]; S.shape_clust_accept=[];
%==========================Gibbs sampling start===================
for it = 1:(burn_in+nr_iterations)
    progressDisplay(it,burn_in+nr_iterations)
    zdp = sampleAllocations(minabstand,shape,scale,Pi,gammaClust,gammaNonClust);
    welche2=find(zdp==2);
    zsum=length(welche2);
    pi2 = betarnd(alphamix,betamix);
    Pi = [1-pi2 pi2]';
    alphamix = alphamix0 + zsum;
    betamix = betamix0 + n - zsum;
    %sample indicator y for clustered vs. non-clustered;
    [shape,scale,shapeNonclustAccept,shapeClustAccept,grenze_alloc] =
        sampleShapeScale(minabstand,k,zdp,shape_nonclust_alpha0,
        shape_nonclust_beta0,scale_nonclust_alpha0,scale_nonclust_beta0,
```

```
    shape_clust_alpha0,shape_clust_beta0,scale_clust_alpha0,
    scale_clust_beta0,gammaClust,gammaNonClust,shape,scale,sdShape);
%[shape,scale,shapeNonclustAccept,shapeClustAccept,grenze_alloc] =
    sampleShapeScale(minabstand,k,zdp,shape_nonclust_alpha0,
    shape_nonclust_beta0,scale_nonclust_alpha0,scale_nonclust_beta0,
    shape_clust_alpha0,shape_clust_beta0,scale_clust_alpha0,
    scale_clust_beta0,gammaClust,gammaNonClust,folge,shape,
    scale,MH_factor);
y = zdp-1;
welche = find(y == 1);
xakt = x(welche,:);
nakt = sum(y == 1);
cut_folge1 = 0:0.001:max(minabstand);
if(weighted_cutoff == 1 | 2)
%%%%######################## WITH WEIGHTING %%%%####################
pdfwerte0_1_mit = (1-pi2)*gampdf(cut_folge1,shape(1),scale(1));
pdfwerte1_1_mit = pi2*gampdf(cut_folge1,shape(2),scale(2));
pdf_differenz_mit = pdfwerte1_1_mit - pdfwerte0_1_mit;
unter_null_mit = find(pdf_differenz_mit < 0);
ueber_null_mit = find(pdf_differenz_mit > 0);
if length(ueber_null_mit) > 0 && length(unter_null_mit) > 0
 if median(ueber_null_mit) < median(unter_null_mit)
   modus_ind_mit = find(pdfwerte1_1_mit == max(pdfwerte1_1_mit));
    cut_folge2_mit = cut_folge1(modus_ind_mit):0.001:max(minabstand);
    pdfwerte0_2_mit = (1-pi2)*gampdf(cut_folge2_mit,shape(1),scale(1));
    pdfwerte1_2_mit = pi2*gampdf(cut_folge2_mit,shape(2),scale(2));
    pdf_differenz_mit = pdfwerte1_2_mit - pdfwerte0_2_mit;
    unter_null_mit = find(pdf_differenz_mit < 0);
    grenze_mit = cut_folge2_mit(unter_null_mit(1));
    ind_transfer_mit = find(abs(minabstand - grenze_mit) ==
        min(abs(minabstand - grenze_mit)));
    grenze_mit = min(minabstand_1(ind_transfer_mit));
 else
    modus_ind_mit = find(pdfwerte0_1_mit == max(pdfwerte0_1_mit));
    cut_folge2_mit = cut_folge1(modus_ind_mit):0.001:max(minabstand);
    pdfwerte0_2_mit = (1-pi2)*gampdf(cut_folge2_mit,shape(1),scale(1));
```

```
    pdfwerte1_2_mit = pi2*gampdf(cut_folge2_mit,shape(2),scale(2));
    pdf_differenz_mit = pdfwerte0_2_mit - pdfwerte1_2_mit;
    ueber_null_mit = find(pdf_differenz_mit > 0);
    grenze_mit = cut_folge2_mit(ueber_null_mit(1));
    ind_transfer_mit = find(abs(minabstand - grenze_mit) ==
        min(abs(minabstand - grenze_mit)));
    grenze_mit = min(minabstand_1(ind_transfer_mit));
 end
end
if length(ueber_null_mit) < 1 || length(unter_null_mit) < 1
    grenze_mit = grenze0;
end
end
if(weighted_cutoff == 0 | 2)
%%%%################### WITHOUT WEIGHTING %%%%##############################
pdfwerte0_1_ohne = gampdf(cut_folge1,shape(1),scale(1));
pdfwerte1_1_ohne = gampdf(cut_folge1,shape(2),scale(2));
pdf_differenz_ohne = pdfwerte1_1_ohne - pdfwerte0_1_ohne;
unter_null_ohne = find(pdf_differenz_ohne < 0);
ueber_null_ohne = find(pdf_differenz_ohne > 0);
if length(ueber_null_ohne) > 0 && length(unter_null_ohne) > 0
 if median(ueber_null_ohne) < median(unter_null_ohne)
   modus_ind_ohne = find(pdfwerte1_1_ohne == max(pdfwerte1_1_ohne));
    cut_folge2_ohne = cut_folge1(modus_ind_ohne):0.001:max(minabstand);
    pdfwerte0_2_ohne = gampdf(cut_folge2_ohne,shape(1),scale(1));
    pdfwerte1_2_ohne = gampdf(cut_folge2_ohne,shape(2),scale(2));
    pdf_differenz_ohne = pdfwerte1_2_ohne - pdfwerte0_2_ohne;
    unter_null_ohne = find(pdf_differenz_ohne < 0);
    grenze_ohne = cut_folge2_ohne(unter_null_ohne(1));
    ind_transfer_ohne = find(abs(minabstand - grenze_ohne) ==
        min(abs(minabstand - grenze_ohne)));
    grenze_ohne = min(minabstand_1(ind_transfer_ohne));
 else
    modus_ind_ohne = find(pdfwerte0_1_ohne == max(pdfwerte0_1_ohne));
    cut_folge2_ohne = cut_folge1(modus_ind_ohne):0.001:max(minabstand);
    pdfwerte0_2_ohne = gampdf(cut_folge2_ohne,shape(1),scale(1));
```

```
    pdfwerte1_2_ohne = gampdf(cut_folge2_ohne,shape(2),scale(2));
    pdf_differenz_ohne = pdfwerte0_2_ohne - pdfwerte1_2_ohne;
    ueber_null_ohne = find(pdf_differenz_ohne > 0);
    grenze_ohne = cut_folge2_ohne(ueber_null_ohne(1));
    ind_transfer_ohne = find(abs(minabstand - grenze_ohne) ==
        min(abs(minabstand - grenze_ohne)));
    grenze_ohne = min(minabstand_1(ind_transfer_ohne));
 end
end
if length(ueber_null_ohne) < 1 || length(unter_null_ohne) < 1
    grenze_ohne = grenze0;
end
end
%================ Initialisation ==============================
abstand_cluster = abstand(welche,welche);
welche_dreieck = find(abstand_cluster ~= 0);
%===== CLUSTER RECONSTRUCTION FOR MIXTURE MODEL WITH WEIGHTING ========
if(weighted_cutoff == 1 | 2)
% Find clusters
welche1_mit = find(abstand_cluster(welche_dreieck) <=
    grenze_mit*std_abstand3_1);
% start / ziel describe all possible connections between two proteins
start_mit = mod(welche_dreieck(welche1_mit), nakt);
start_mit(start_mit == 0) = nakt; %Nullen bei Modulo-Ergebnis ersetzen
start_mit(start_mit == 0) = nakt; %Nullen bei Modulo-Ergebnis ersetzen
ziel_mit = ceil(welche_dreieck(welche1_mit)/nakt);
cluster_elemente_mit = cell(1,1);
cluster_abstand_mit = cell(1,1);
for m = 1:length(start_mit)
    index_start_mit = 0;
    index_ziel_mit = 0;
    %Security var cleaning
    wz_idx_mit = [];
    ws_idx_mit = [];
    cumlistOfIdx_mit = [];
    %Analyzes the whole cluster_elemente looking for ziel(m)/start(m)
```

```
%Returns a binary matrix - only matricial index is available because
%of the cell2mat cast
wz_idx_mit = ismember([cluster_elemente_mit{:}],ziel_mit(m));
ws_idx_mit = ismember([cluster_elemente_mit{:}],start_mit(m));
%Apply length on all cell members from cluster_elemente
%Do cumsum on it to get correspondance between mat indx and cell idx
cumlistOfIdx_mit = cumsum(cellfun('length',cluster_elemente_mit));
    %if true there is a least one match
    if sum(wz_idx_mit) ~= 0
        idx_mit = find(wz_idx_mit == 1);              %get mat idx
        index_ziel_mit = find(cumlistOfIdx_mit >= idx_mit,1,
            'first'); %find corresponding cell idx
    end
    if sum(ws_idx_mit) ~= 0
        idx2_mit = find(ws_idx_mit == 1);
        index_start_mit = find(cumlistOfIdx_mit >= idx2_mit,1,'first');
    end
  if index_start_mit == 0 && index_ziel_mit > 0
     % add Start to Cluster Ziel
     cluster_elemente_mit{index_ziel_mit} =
        [cluster_elemente_mit{index_ziel_mit} start_mit(m)];
     for w = 1:length(cluster_elemente_mit{index_ziel_mit})
        cluster_abstand_mit{index_ziel_mit} =
            [cluster_abstand_mit{index_ziel_mit}
            abstand_cluster(start_mit(m),
            cluster_elemente_mit{index_ziel_mit}(w))];
     end
  else
     if index_start_mit > 0 && index_ziel_mit == 0
        % add Ziel to Cluster Start
        cluster_elemente_mit{index_start_mit} =
            [cluster_elemente_mit{index_start_mit} ziel_mit(m)];
        for w = 1:length(cluster_elemente_mit{index_start_mit})
            cluster_abstand_mit{index_start_mit} =
                [cluster_abstand_mit{index_start_mit}
                abstand_cluster(cluster_elemente_mit
```

```
                    {index_start_mit}(w),ziel_mit(m))];
            end
        else
            if index_start_mit > 0 && index_ziel_mit > 0 &&
                    index_start_mit ~= index_ziel_mit
                % Vereinigung zweier Cluster
                cluster_elemente_mit{index_start_mit} =
                    [cluster_elemente_mit{index_start_mit}
                    cluster_elemente_mit{index_ziel_mit}];
                cluster_elemente_mit = cluster_elemente_mit
                    ([1:(index_ziel_mit-1) (index_ziel_mit+1):end]);
                cluster_abstand_mit{index_start_mit} =
                    [cluster_abstand_mit{index_start_mit}
                    cluster_abstand_mit{index_ziel_mit}];
                cluster_abstand_mit = cluster_abstand_mit
                    ([1:(index_ziel_mit-1) (index_ziel_mit+1):end]);
            else
                if index_start_mit == 0 && index_ziel_mit == 0
                  cluster_abstand_mit{length(cluster_elemente_mit)+1} =
                        abstand_cluster(start_mit(m),ziel_mit(m));
                  cluster_elemente_mit{length(cluster_elemente_mit)+1} =
                        [start_mit(m) ziel_mit(m)];
                end
            end
        end
    end
    ende_mit = length(cluster_elemente_mit);
    cluster_elemente_mit = cluster_elemente_mit(2:ende_mit);
    cluster_abstand_mit = cluster_abstand_mit(2:ende_mit);
    % remove clusters which partially lie in the border to avoid bias
    for q = 1:length(cluster_elemente_mit)
        if q <= length(cluster_elemente_mit)
        elemente_mit = cluster_elemente_mit{q};
        infrage_mit = abstand_temp(final_ind_complement,elemente_mit);
         if min(min(infrage_mit)) <= grenze_mit*std_abstand3_1
```

```
        cluster_elemente_mit = cluster_elemente_mit([1:(q-1) (q+1):end]);
        cluster_abstand_mit = cluster_abstand_mit([1:(q-1) (q+1):end]);
     end
    end
end
%sort elements in clusters and count elements in all clusters
ssize_mit = [];
cluster_max_abstand_mit = [];
cluster_max_abstand1_mit = [];
for q = 1:length(cluster_elemente_mit)
    cluster_elemente_mit{q} = sort(cluster_elemente_mit{q});
    cluster_max_abstand_mit = [cluster_max_abstand_mit (2/(128/(45*pi)))*
        mean(sqrt(cluster_abstand_mit{q}))];  % also reverse squaring
cluster_max_abstand1_mit = [cluster_max_abstand1_mit (2/(128/(45*pi)))*
        median(sqrt(cluster_abstand_mit{q}))];
ssize_mit = [ssize_mit length(cluster_elemente_mit{q})];
end
nr_clusters_mit = length(cluster_elemente_mit);
geclusterte_mit = [];
for q = 1:length(cluster_elemente_mit)
    geclusterte_mit = [geclusterte_mit sort(cluster_elemente_mit{q})];
end
% calculation of measures
prop_clust_mit = sum(ssize_mit)/n;
size_mean_mit = mean(ssize_mit);
radius_mean_mit = mean(cluster_max_abstand_mit)/2;
radius_mean1_mit = mean(cluster_max_abstand1_mit)/2;
if length(ssize_mit) == 0
     size_qx_mit = repmat(0,1,length(quantilevector));
     radius_qx_mit = repmat(0,1,length(quantilevector));
else
     size_qx_mit = quantile(ssize_mit,quantilevector);
     radius_qx_mit = quantile(cluster_max_abstand_mit,quantilevector)/2;
end
end
if(weighted_cutoff == 0 | 2)
```

```
%=== CLUSTER RECONSTRUCTION FOR MIXTURE MODEL WITHOUT WEIGHTING ========
% Find clusters
welche1_ohne = find(abstand_cluster(welche_dreieck) <=
    grenze_ohne*std_abstand3_1);
% start / ziel describe all possible connections between two proteins
start_ohne = mod(welche_dreieck(welche1_ohne), nakt);
start_ohne(start_ohne == 0) = nakt; %Nullen bei Modulo-Ergebnis ersetzen
start_ohne(start_ohne == 0) = nakt; %Nullen bei Modulo-Ergebnis ersetzen
ziel_ohne = ceil(welche_dreieck(welche1_ohne)/nakt);
cluster_elemente_ohne = cell(1,1);
cluster_abstand_ohne = cell(1,1);
for m = 1:length(start_ohne)
    index_start_ohne = 0;
    index_ziel_ohne = 0;
    %Security var cleaning
    wz_idx_ohne = [];
    ws_idx_ohne = [];
    cumlistOfIdx_ohne = [];
    %Analyzes the whole cluster_elemente looking for ziel(m)/start(m)
    %Returns a binary matrix - only matricial index is available
    %because of the cell2mat cast
    wz_idx_ohne = ismember([cluster_elemente_ohne{:}],ziel_ohne(m));
    ws_idx_ohne = ismember([cluster_elemente_ohne{:}],start_ohne(m));
    %Apply length on all cell members from cluster_elemente
    %Do cumsum on it to get correspondance between mat indx and cell idx
    cumlistOfIdx_ohne = cumsum(cellfun('length',cluster_elemente_ohne));
        %if true there is a least one match
        if sum(wz_idx_ohne) ~= 0
            idx_ohne = find(wz_idx_ohne == 1);    %get mat idx
            index_ziel_ohne = find(cumlistOfIdx_ohne >= idx_ohne,1,
                'first'); %find corresponding cell idx
        end
        if sum(ws_idx_ohne) ~= 0
            idx2_ohne = find(ws_idx_ohne == 1);
            index_start_ohne = find(cumlistOfIdx_ohne >= idx2_ohne,1,'first');
        end
```

```
if index_start_ohne == 0 && index_ziel_ohne > 0
   % add Start to Cluster Ziel
   cluster_elemente_ohne{index_ziel_ohne} =
      [cluster_elemente_ohne{index_ziel_ohne} start_ohne(m)];
   for w = 1:length(cluster_elemente_ohne{index_ziel_ohne})
      cluster_abstand_ohne{index_ziel_ohne} =
          [cluster_abstand_ohne{index_ziel_ohne}
          abstand_cluster(start_ohne(m),
          cluster_elemente_ohne{index_ziel_ohne}(w))];
   end
else
   if index_start_ohne > 0 && index_ziel_ohne == 0
      % add Ziel to Cluster Start
      cluster_elemente_ohne{index_start_ohne} =
          [cluster_elemente_ohne{index_start_ohne} ziel_ohne(m)];
      for w = 1:length(cluster_elemente_ohne{index_start_ohne})
          cluster_abstand_ohne{index_start_ohne} =
              [cluster_abstand_ohne{index_start_ohne}
               abstand_cluster(cluster_elemente_ohne
                  {index_start_ohne}(w),ziel_ohne(m))];
      end
   else
      if index_start_ohne > 0 && index_ziel_ohne > 0 &&
      index_start_ohne ~= index_ziel_ohne
         % Vereinigung zweier Cluster
         cluster_elemente_ohne{index_start_ohne} =
              [cluster_elemente_ohne{index_start_ohne}
              cluster_elemente_ohne{index_ziel_ohne}];
         cluster_elemente_ohne = cluster_elemente_ohne(
            [1:(index_ziel_ohne-1) (index_ziel_ohne+1):end]);
         cluster_abstand_ohne{index_start_ohne} =
              [cluster_abstand_ohne{index_start_ohne}
              cluster_abstand_ohne{index_ziel_ohne}];
         cluster_abstand_ohne = cluster_abstand_ohne(
              [1:(index_ziel_ohne-1) (index_ziel_ohne+1):end]);
      else
```

```
                if index_start_ohne == 0 && index_ziel_ohne == 0
                  cluster_abstand_ohne{length(cluster_elemente_ohne)+1} =
                         abstand_cluster(start_ohne(m),ziel_ohne(m));
                  cluster_elemente_ohne{length(cluster_elemente_ohne)+1} =
                         [start_ohne(m) ziel_ohne(m)];
                end
              end
           end
        end
end
ende_ohne = length(cluster_elemente_ohne);
cluster_elemente_ohne = cluster_elemente_ohne(2:ende_ohne);
cluster_abstand_ohne = cluster_abstand_ohne(2:ende_ohne);
% remove clusters which partially lie in the border to avoid bias
for q = 1:length(cluster_elemente_ohne)
    if q <= length(cluster_elemente_ohne)
    elemente_ohne = cluster_elemente_ohne{q};
    infrage_ohne = abstand_temp(final_ind_complement,elemente_ohne);
     if min(min(infrage_ohne)) <= grenze_ohne*std_abstand3_1
        cluster_elemente_ohne = cluster_elemente_ohne([1:(q-1) (q+1):end]);
        cluster_abstand_ohne = cluster_abstand_ohne([1:(q-1) (q+1):end]);
     end
    end
end
%sort elements in clusters and count elements in all clusters
ssize_ohne = [];
cluster_max_abstand_ohne = [];
cluster_max_abstand1_ohne = [];
for q = 1:length(cluster_elemente_ohne)
    cluster_elemente_ohne{q} = sort(cluster_elemente_ohne{q});
    cluster_max_abstand_ohne = [cluster_max_abstand_ohne (2/(128/(45*pi)))*
        mean(sqrt(cluster_abstand_ohne{q}))];      % also reverse squaring
    cluster_max_abstand1_ohne = [cluster_max_abstand1_ohne
        (2/(128/(45*pi)))*median(sqrt(cluster_abstand_ohne{q}))];
    ssize_ohne = [ssize_ohne length(cluster_elemente_ohne{q})];
end
```

```
nr_clusters_ohne = length(cluster_elemente_ohne);
geclusterte_ohne = [];
for q = 1:length(cluster_elemente_ohne)
    geclusterte_ohne = [geclusterte_ohne sort(cluster_elemente_ohne{q})];
end
% calculation of measures
prop_clust_ohne = sum(ssize_ohne)/n;
size_mean_ohne = mean(ssize_ohne);
radius_mean_ohne = mean(cluster_max_abstand_ohne)/2;
radius_mean1_ohne = mean(cluster_max_abstand1_ohne)/2;
if length(ssize_ohne) == 0
     size_qx_ohne = repmat(0,1,length(quantilevector));
     radius_qx_ohne = repmat(0,1,length(quantilevector));
else
     size_qx_ohne = quantile(ssize_ohne,quantilevector);
     radius_qx_ohne = quantile(cluster_max_abstand_ohne,quantilevector)/2;
end
end
%==================== iterationwise misclassification rate ============
y_alles = zeros(n_alles,1);
y_alles(keine_ausreisser) = y;
if simulated == 1
    non_clustNotInBorder = setdiff(NotInBorder, clustNotInBorder);
    tp=length(intersect(clustNotInBorder, NotInBorder(y_alles==1)));
    fp=length(intersect(non_clustNotInBorder,NotInBorder(y_alles==1)));
    tn=length(intersect(non_clustNotInBorder,NotInBorder(y_alles==0)));
    fn=n_alles-tp-fp-tn;
    mcr=(fp+fn)/n_alles;
end
%==================== store iterations ============================
if it > burn_in && mod(it,thin) == 0
    S.prop_distr=[S.prop_distr Pi(2)];
    S.prop=[S.prop sum(y_alles)/n_alles];
    if(weighted_cutoff == 1)
      S.size_mit=[S.size_mit; size_mean_mit];
   S.size_qx_mit=[S.size_qx_mit; size_qx_mit];
```

```
    S.radius_mit=[S.radius_mit; radius_mean_mit];
    S.radius1_mit=[S.radius1_mit; radius_mean1_mit];
    S.radius_qx_mit=[S.radius_qx_mit; radius_qx_mit];
    S.nr_clusters_mit=[S.nr_clusters_mit nr_clusters_mit];
    S.grenze_mit=[S.grenze_mit (grenze_mit*std_abstand3_1)];
  else
    S.size_ohne=[S.size_ohne; size_mean_ohne];
    S.size_qx_ohne=[S.size_qx_ohne; size_qx_ohne];
    S.radius_ohne=[S.radius_ohne; radius_mean_ohne];
    S.radius1_ohne=[S.radius1_ohne; radius_mean1_ohne];
    S.radius_qx_ohne=[S.radius_qx_ohne; radius_qx_ohne];
    S.nr_clusters_ohne=[S.nr_clusters_ohne nr_clusters_ohne];
    S.grenze_ohne=[S.grenze_ohne (grenze_ohne*std_abstand3_1)];
  end
     S.y_alles{it}=y_alles;
     if simulated == 1
         S.mcr=[S.mcr mcr];
     end
     S.nakt=[S.nakt nakt];
     S.scale_nonclust=[S.scale_nonclust scale(1)];
         S.scale_clust=[S.scale_clust scale(2)];
     S.shape_nonclust=[S.shape_nonclust shape(1)];
         S.shape_clust=[S.shape_clust shape(2)];
     S.shape_nonclust_accept=[S.shape_nonclust_accept shapeNonclustAccept];
     S.shape_clust_accept=[S.shape_clust_accept shapeClustAccept];
end
S.shape_nonclust_accept =
     S.shape_nonclust_accept(S.shape_nonclust_accept >= 0);
S.shape_clust_accept = S.shape_clust_accept(S.shape_clust_accept >= 0);
S.minabstand = minabstand; S.scaling = scaling_distances;
S.minabstand_temp = min(abstand_temp);
S.minabstand_temp2 = minabstand_temp2;
S.std_abstand3 = std_abstand3; std_abstand3_1 = std_abstand3_1;
S.sdShape = sdShape;
S.detection_error = detection_error; S.knn = knn;
if simulated == 1
```

```
      S.proportion_clustered = proportion_clustered;
      S.mean_cluster_radius = mean_cluster_radius;
      S.mean_cluster_size = mean_cluster_size;
      S.proportion_clustered_in = proportion_clustered_in;
      S.proportion_within_metaclusters_in = proportion_within_metaclusters_in;
      S.mean_cluster_radius_in = mean_cluster_radius_in;
      S.mean_cluster_size_in = mean_cluster_size_in;
      S.size_type_in = size_type_in;
      S.detection_error_sd_in = detection_error_sd;
      S.transform_range_x_in = transform_range_x_in;
      S.transform_range_y_in = transform_range_y_in;
      S.nr_first_parents_in = nr_first_parents_in;
      %S.dark_fraction_in = dark_fraction_in;
      S.eps_in = eps_in;
      S.proportion_points_replaced_with_cluster_in =
            proportion_points_replaced_with_cluster;
      S.proportion_points_replaced_with_metacluster_in =
            proportion_points_replaced_with_metacluster_in;
      S.point_density_in = point_density_in;
      S.nr_points_in = nr_points_in;
      S.clust = clust;
      S.final_ind = final_ind;
      S.NotInBorder = NotInBorder;
    end
    S.shape_nonclust_alpha0= shape_nonclust_alpha0;
    S.shape_nonclust_beta0= shape_nonclust_beta0;
    S.scale_nonclust_alpha0= scale_nonclust_alpha0;
    S.scale_nonclust_beta0= scale_nonclust_beta0;
    S.shape_clust_alpha0= shape_clust_alpha0;
    S.shape_clust_beta0= shape_clust_beta0;
    S.scale_clust_alpha0=scale_clust_alpha0;
    S.scale_clust_beta0= scale_clust_beta0;
end
toc
```

C.3.2 Auxiliary Functions

Function `progressDisplay`

This function was supplied by Thomas Klein, based on code by Adam Leadbetter (see *http://www.mathworks.com/matlabcentral/fileexchange/24099-percent-done/content/perccount.m*, last accessed on 19.08.2014)

```
function progressDisplay(jj,maxjj)

%Reports the percentage of the job done to the screen.
% PERCOUNT(I,Imax)
% I is the current iteration between 1 and Imax
% Imax is the maximum number of iterations
% Do not print anything to the screen between calls of this function!
% title - s cmspike/perccount vr - 1.2
% author - bodc/alead date - 2006may16
% Updated 2009May14
% At suggestion of John D'Errico renamed internal variable "max" to
% "maxjj"
% Also following D'Errico's suggestions the following functionality has
% been added:
% 1. An invocation check - checks that two input arguments are supplied

% lastCall is the last integer percent displayed persistent lastCall;
% if the number of arguments is 2
if(nargin == 2)
% if this is the first run, make lastCall a value that will get the
% following if loops working from the start.
if isempty(lastCall)
lastCall = -1;
end
% if the percentage has increased to the next integer, then the
% displayed percentage will need to be updated
if(lastCall ~= floor(((jj-1)/maxjj) * 100))
% if the progress so far (jj out of maxjj) is not the first
% representation, then backspace the percentage area to prepare for
```

```
% the next percentage update.
if(jj ~= 1)
fprintf(1,'\b\b\b');
% if the progress so far IS the first time it is displayed, then
% display the "context line" of the percentage.
else
fprintf(1,'\n\tPercentage complete: ');
end
% Calculate the percentage done
pc_done = num2str(floor(((jj-1)/maxjj) * 100));
% if the percentage done so far is in the single digit range
% (between 0 and 10), then add a zero before it so the percentage
% displays neatly
if(length(pc_done) == 1)
pc_done(2) = pc_done(1);
pc_done(1) = '0';
end
fprintf(1,'%s%%',pc_done);
end
lastCall = floor(((jj-1)/maxjj) * 100); % the last int percent displayed
% if this is the last part of the total, then display 100% complete and
% add a few lines of whitespace for neatness
if(jj == maxjj)
fprintf(1,'\b\b\b100%%\n\n');
end
else
error('Error: PERCCOUNT needs two input arguments...');
end
```

Function `sampleAllocations`

```
function z = sampleAllocations(x,shape,scale,Pi,gammaClust,gammaNonClust)
n = length(x);
P1=Pi;P2=zeros(2,n);
for j=gammaNonClust
    P2(j,:)=gampdf(x',shape(j),scale(j))';
end
for j=gammaClust
    P2(j,:)=gampdf(x',shape(j),scale(j))';
end
P=repmat(P1,1,n);P=P.*P2;
P=P./repmat(sum(P,1),2,1);
%Resampling Index z for i=1,...,n
Prop_n=cumsum(P);
rn=rand(1,n);  z=zeros(1,n);
ir=find(rn<=Prop_n(1,:));
if ~isempty(ir)
    z(ir)=1;
end
for j=2
    ir=find(rn>=Prop_n(j-1,:)&rn<Prop_n(j,:));
    if ~isempty(ir)
        z(ir)=j; end
end
nz=zeros(2,1);
for j=1:2
    i=find(z==j); if ~isempty(i) nz(j)=length(i); end
end
kc=length(find(nz~=0));
```

Function sampleShapeScale

```
function [shape,scale,shapeNonclustAccept,shapeClustAccept,grenze] =
    sampleShapeScale(werte,kmax,zdp,shape_nonclust_alpha0,shape_nonclust_beta0,
    scale_nonclust_alpha0,scale_nonclust_beta0,shape_clust_alpha0,
    shape_clust_beta0, scale_clust_alpha0,scale_clust_beta0,gammaClust,
    gammaNonClust,shape,scale,sdShape)
% Initialize return objects
minwert=[];maxwert=[];
for j=1:kmax
    i=find(zdp==j); nj=length(i);
    % if component is not empty
    if (nj>0)
        % Update of distributions
        if length(intersect(j, gammaNonClust)) == 1
        % Gamma distributions for non-clustered points
            werteNonclust = werte(i);
            minwert = min(werteNonclust);
            xsum_nonclust = sum(werteNonclust);
            xprod_nonclust = prod(werteNonclust);
            n_nonclust = length(werteNonclust);
            % Updates
            scale_nonclust_alpha = shape(1)*n_nonclust + scale_nonclust_alpha0;
            scale_nonclust_beta = scale_nonclust_beta0/
                (1+scale_nonclust_beta0*xsum_nonclust);
        elseif length(intersect(j, gammaClust)) == 1
            % Gamma distributions for clustered points
            werteClust = werte(i);
            maxwert = max(werteClust);
            xsum_clust = sum(werteClust);
            xprod_clust = prod(werteClust);
            n_clust = length(werteClust);
            % Updates
            scale_clust_alpha = shape(2)*n_clust + scale_clust_alpha0;
            scale_clust_beta = scale_clust_beta0/
                (1+scale_clust_beta0*xsum_clust);
```

```
        end
        % draw from updated distributions
        if length(intersect(j, gammaNonClust)) == 1
        % draw values from Gamma distributions for non-clustered points
          scale(j) = 1/gamrnd(scale_nonclust_alpha,scale_nonclust_beta);
          [shape(j),shapeNonclustAccept]=sampleShape(shape(1),scale(1),
                n_nonclust,shape_nonclust_alpha0,
                shape_nonclust_beta0,werteNonclust,sdShape);
        elseif length(intersect(j, gammaClust)) == 1
        % draw values from Gamma distributions for clustered points
          scale(j) = 1/gamrnd(scale_clust_alpha,scale_clust_beta);
          [shape(j),shapeClustAccept] = sampleShape(shape(2),scale(2),
            n_clust,shape_clust_alpha0,
            shape_clust_beta0,werteClust,sdShape);
        end
   else
   % if component is empty: draw from prior
        if length(intersect(j, gammaNonClust)) == 1
        % Gamma distributions for non-clustered points
          shape(j) = gamrnd(shape_nonclust_alpha0,shape_nonclust_beta0);
          scale(j) = 1/gamrnd(scale_nonclust_alpha0,scale_nonclust_beta0);
   shapeNonclustAccept = -1;
        elseif length(intersect(j, gammaClust)) == 1
       % Gamma distributions for clustered points
          shape(j) = gamrnd(shape_clust_alpha0,shape_clust_beta0);
          scale(j) = 1/gamrnd(scale_clust_alpha0,scale_clust_beta0);
   shapeClustAccept = -1;
        end
    end
    grenze = mean([maxwert minwert]);
end
```

Function sampleShape

```
function [alpha_neu,alpha_accept]= sampleShape(alpha_alt,scale,n,a,b,werte,
    sdShape)
%############## pi_Y #################
likelihood_alt = sum(log(gampdf(werte,alpha_alt*ones(1,n),scale*ones(1,n))));
alpha_alt_prior = log(gampdf(alpha_alt,a,b));
pi_X_log = likelihood_alt+alpha_alt_prior;
%############################ Proposal q(Y|X) ############################
alpha_neu_prop = abs(normrnd(alpha_alt,sdShape));
q_yx_log = log(normpdf(alpha_neu_prop,alpha_alt,sdShape));
%############## pi_Y #################
likelihood_neu = sum(log(gampdf(werte,alpha_neu_prop*ones(1,n),
    scale*ones(1,n))));
alpha_neu_prior = log(gampdf(alpha_neu_prop,a,b));
pi_Y_log = likelihood_neu+alpha_neu_prior;
%######### Proposal q(X|Y) #############
alpha_alt_prop = abs(normrnd(alpha_neu_prop,sdShape));
q_xy_log = log(normpdf(alpha_alt_prop,alpha_neu_prop,sdShape));
%% Acceptance step
shapeAlpha_akzep = min(0, pi_Y_log+q_xy_log-pi_X_log-q_yx_log);
u = log(unifrnd(0,1));
if ~isnan(shapeAlpha_akzep) && u < shapeAlpha_akzep
    alpha_neu = alpha_neu_prop;
    alpha_accept = 1;
else
    alpha_neu = alpha_alt;
    alpha_accept = 0;
end
```

C.3.3 Example Code

Execution of GAMMICS code

This is an exemplary MATLAB work flow to run the GAMMICS method. Adaptions may be necessary for new analyses.

```
% set up working directory in which all files are located
cd /home/GAMMICS/;
% clear work space
clear all; close all;
% load data
load simulated_cell_coordinates.mat;
% Are the data simulated or not? (experimental=0, simulated = 1)
simulated = 1;
% Is a detection error simulated? (only relevant for simulated data; no=0, yes=1)
detection_error = 1;
% Specify the upper and lower quantiles used for the definition of outliers
upper_quantile = 0.975;
lower_quantile = 0.025;
% Specify the factor to be multiplied with the MAD of distances to obtain the
% normalization constant for scaling the distances
scaling_distances = 2;
% Specify the limits of the quadratic analysed region of interest
% (coordinates of left, right, bottom and top limits)
left =  0; right =  4000;
bottom = 0; top =  4000;
% Specify the border width for the correction of edge effects
borderWidth = 100;
% MCMC configuration: number of iterations, number of iterations used as
% burn-in, thinning
nr_iterations = 22500;
burn_in = 500;
thin    = 10;
% Specify the quantiles across clusters to keep trace of for distributions
% of cluster size and cluster radius
quantilevector = 0:0.01:1;
```

```
% For which degree of nearest neighbors should distances be calculated for
% the model? (e.g., knn=2 results in distances to the second nearest neighbors)
knn = 2;
% Specify the standard deviation of the proposal distribution in the MH step
% for the gamma shape distributions
sdShape=0.1;
% Should the gamma distributions be weighted in the calculation of the cutoff
% L_c? (0=no, 1=yes, 2=both)
weighted_cutoff = 1;
% Initial values for the gamma distributions
shape_nonclust_init= 3; scale_nonclust_init= 2;
shape_clust_init= 2; scale_clust_init= 0.1;
% do not change
shape = [shape_nonclust_init shape_clust_init];
scale = [scale_nonclust_init scale_clust_init];
% Initial values for the beta distribution for p_c
alphamix = 1;
betamix = 1;
% do not change
n = length(X_nm);
Pi=rand(2,1); Pi=Pi/sum(Pi); zdp=rdisc(n,Pi);
% Specify the hyperprior gamma distributions
shape_nonclust_alpha0= 3; shape_nonclust_beta0= 1;
scale_nonclust_alpha0= 1; scale_nonclust_beta0= 0.5;
shape_clust_alpha0= 2; shape_clust_beta0= 1;
scale_clust_alpha0= 10; scale_clust_beta0= 1;
% do not change
shape0 = [shape_nonclust_alpha0*shape_nonclust_beta0
    1/shape_clust_alpha0*shape_clust_beta0];
scale0 =[scale_nonclust_alpha0*scale_nonclust_beta0
    1/scale_clust_alpha0*scale_clust_beta0];
% Specify the hyperprior beta distribution for p_c
alphamix0 = 1;
betamix0 = 1;
% do not change
gammaNonClust = 1;
```

```
gammaClust = 2;
% Call the main routine (takes some time)
GAMMICS;
% Save the results
save('results.mat','S')
```

Analysis of results in R

```
########################## Set working directory ########################
setwd("C://")
#### read in GAMMICS results from MATLAB (requires R.matlab package)####
library(R.matlab)
results <- readMat('results.mat')
###### Extract and label results object ################################
S <- results$S
names(S) <- attr(results$S,"dimnames")[[1]]
########## Define iterations of Markov Chains to be analysed ##########
burn_in = 750;        # define burn-in for analysis (additional to MATLAB)
nr_iterations = 1500; # define number of iterations of chains to be analysed
thin = 3;             # define thinning for analysis (additional to MATLAB)
########################## Examine trace plots ########################
x = seq(burn_in+1,burn_in+nr_iterations,thin);
plot(1:length(x),S$shape.nonclust[x], type="l")
plot(1:length(x),S$shape.clust[x], type="l")
plot(1:length(x),S$scale.nonclust[x], type="l")
plot(1:length(x),S$scale.clust[x], type="l")
plot(1:length(x),S$prop[x], type="l")
###### weighted gamma distributions when calculating cutoff ###############
plot(1:length(x),S$size.mit[x], type="l")        # mean cluster size
plot(1:length(x),S$radius.mit[x], type="l")      # mean cluster radius
plot(1:length(x),S$nr.clusters.mit[x], type="l") # number of clusters
plot(1:length(x),sqrt(S$grenze.mit[x]), type="l")      # cutoff
###### unweighted gamma distributions when calculating cutoff #############
plot(1:length(x),S$size.ohne[x], type="l")        # mean cluster size
plot(1:length(x),S$radius.ohne[x], type="l")      # mean cluster radius
plot(1:length(x),S$nr.clusters.ohne[x], type="l") # number of clusters
```

```
plot(1:length(x),sqrt(S$grenze.ohne[x]), type="l")      # cutoff
#### Examine acceptance ratio in Matropolis-Hastings step #############
mean(S$shape.nonclust.accept[x])
mean(S$shape.clust.accept[x])
######## Examine autocorrelations (requires fBasics package) #########
library(fBasics)
acfPlot(S$shape.nonclust[x])
acfPlot(S$shape.clust[x])
acfPlot(S$scale.nonclust[x])
acfPlot(S$scale.clust[x])
acfPlot(S$prop[x])
############## View histograms across clusters ######################
######weighted gamma distributions when calculating cutoff #############
sizes_mit <- matrix(S$size.qx.mit[x,],ncol=101)
radii_mit <- matrix(S$radius.qx.mit[x,],ncol=101)
size_quantiles_mit = apply(sizes_mit,2,median);
radius_quantiles_mit = apply(radii_mit,2,median);
par(mfrow=c(1,2))
par(mar=c(4,4,4,2)+0.1)
hist(size_quantiles_mit, xlab="size")
par(mar=c(4,8,4,2)+0.1)
hist(radius_quantiles_mit, xlab="radius (nm)")
###### unweighted gamma distributions when calculating cutoff ##########
sizes_ohne <- matrix(S$size.qx.ohne[x,],ncol=101)
radii_ohne <- matrix(S$radius.qx.ohne[x,],ncol=101)
size_quantiles_ohne = apply(sizes_ohne ,2,median);
radius_quantiles_ohne = apply(radii_ohne,2,median);
par(mfrow=c(1,2))
par(mar=c(4,4,4,2)+0.1)
hist(size_quantiles_ohne, xlab="size")
par(mar=c(4,8,4,2)+0.1)
hist(radius_quantiles_ohne, xlab="radius (nm)")
####### View histograms across MCMC iterations ####################
############## weighted gamma distributions when calculating cutoff
par(mfrow=c(3,2))
####### p_c #######
```

```
hist(S$prop[x], main="", xlab=expression(p[c]))
####### cutoff #######
hist(sqrt(S$grenze.mit[x]), main="",
    xlab=expression(paste(tilde(d)[c]," (nm)")))
####### mu_c #######
hist(S$size.mit[x], main="",
    xlab=expression(paste("mean size (",mu[c]," )")))
####### r_c #######
hist(S$radius.mit[x], main="",
    xlab=expression(paste("mean radius (",r[c]," ) (nm)")))
####### number of clusters #######
hist(S$nr.clusters.mit[x], main="", xlab="number of clusters",
    ylab="density", freq=FALSE, breaks = "Scott", right=FALSE)
######## weighted gamma distributions when calculating cutoff ######
par(mfrow=c(3,2))
####### p_c #######
hist(S$prop[x], main="", xlab=expression(p[c]))
####### cutoff #######
hist(sqrt(S$grenze.ohne[x]), main="",
    xlab=expression(paste(tilde(d)," (nm)")))
####### mu_c #######
hist(S$size.ohne[x], main="",
    xlab=expression(paste("mean size (",mu[c]," )")))
####### r_c #######
hist(S$radius.ohne[x], main="",
    xlab=expression(paste("mean radius (",r[c]," ) (nm)")))
####### number of clusters #######
hist(S$nr.clusters.ohne[x], main="", xlab="number of clusters")
############### View parameter estimates ######################
# proportion of clustered proteins (median across chain)
median(S$prop[x], na.rm=TRUE)
####### weighted gamma distributions when calculating cutoff ######
##### cluster size ############################################
# mean cluster size (median across chain)
median(S$size.mit[x],na.rm=TRUE)
######################################### cluster radius #########
```

```
# mean cluster radius (median across chain)
median(S$radius.mit[x],na.rm=TRUE)
############################## number of clusters #################
# median number of clusters
median(S$nr.clusters.mit[x], na.rm=TRUE)
############################## cutoff value ######################
# median cutoff for nearest neighbor distances
median(sqrt(S$grenze.mit[x]), na.rm=TRUE)
################# unweighted gamma distributions when calculating cutoff ####
################# cluster size ###############
# mean cluster size (mean and median across chain)
median(S$size.ohne[x],na.rm=TRUE)
########### cluster radius ########################
# mean cluster radius (mean and median across chain)
median(S$radius.ohne[x],na.rm=TRUE)
####################### number of clusters ###########
# mean and median number of clusters
median(S$nr.clusters.ohne[x], na.rm=TRUE)
########################## cutoff value ########################
# mean and median cutoff for nearest neighbor distances
median(sqrt(S$grenze.ohne[x]), na.rm=TRUE)
```

C.4 WinBUGS Code

```
model
{
for (i in 1:N)
{
Z[i] ~ dnorm(muR[i],tauR[i] )
muR[i] <- mu[T[i]]
tauR[i] <- tau[T[i]]
T[i] ~ dcat(pi[])
T1[i] <- equals(T[i],1)
T2[i] <- equals(T[i],2)
T3[i] <- equals(T[i],3)
T4[i] <- equals(T[i],4)
T5[i] <- equals(T[i],5)
T6[i] <- equals(T[i],6)
T7[i] <- equals(T[i],7)
T8[i] <- equals(T[i],8)
}

pi[1:8]~ddirch(alpha[])

mu[1]~dnorm(-10,1)I(a,0.0)
mu[2]~dnorm(-10,1)I(a,0.0)
mu[3]~dnorm(-10,1)I(a,0.0)
mu[4]<-0
mu[5]<-0
mu[6]~dnorm(10,1)I(0.0,b)
mu[7]~dnorm(10,1)I(0.0,b)
mu[8]~dnorm(10,1)I(0.0,b)
tau[1]~dgamma(0.1,0.1)
tau[2]~dgamma(0.1,0.1)
tau[3]~dgamma(0.1,0.1)
tau[4]~dgamma(0.1,0.1)
tau[5]~dgamma(0.1,0.1)
```

```
tau[6]~dgamma(0.1,0.1)
tau[7]~dgamma(0.1,0.1)
tau[8]~dgamma(0.1,0.1)
}
```

C.5 R/Bioconductor Package Epigenomix

This package can be downloaded and installed in `R` by calling

```
> source("http://bioconductor.org/biocLite.R")
> biocLite("epigenomix")
```

In the following, functions of this package that were authored or co-authored by the author of this thesis are documented.

`bayesMixModel` *Fits a Bayesian mixture model using MCMC methods*

Description

This method estimates the posterior distribution of a Bayesian mixture model using Markov Chain Monte Carlo (MCMC) methods and calculates measures of this distribution. The mixture model may consist of normal components (with a fixed expectation of 0), exponential components and gamma components, which may be mirrored in order to model negative values.

Usage

```
bayesMixModel(z, normNull=c(), expNeg=c(), expPos=c(), gamNeg=c(),
    gamPos=c(), sdNormNullInit=c(), rateExpNegInit=c(), rateExpPosInit=c(),
    shapeGamNegInit=c(), scaleGamNegInit=c(), shapeGamPosInit=c(),
    scaleGamPosInit=c(), piInit, classificationsInit, dirichletParInit=1,
    shapeDir=1, scaleDir=1, weightsPrior="FDD", sdAlpha, shapeNorm0=c(),
    scaleNorm0=c(), shapeExpNeg0=c(), scaleExpNeg0=c(), shapeExpPos0=c(),
    scaleExpPos0=c(), shapeGamNegAlpha0=c(), shapeGamNegBeta0=c(),
    scaleGamNegAlpha0=c(), scaleGamNegBeta0=c(), shapeGamPosAlpha0=c(),
    shapeGamPosBeta0=c(), scaleGamPosAlpha0=c(), scaleGamPosBeta0=c(), itb,
    nmc, thin, average="mean",sdShape)
```

Arguments

`z`	Observed values

`normNull`	Indices of the normal components (centered at 0).
`expNeg`	Indices of the mirrored exponential components.
`expPos`	Indices of the exponential components.
`gamNeg`	Indices of the mirrored gamma components.
`gamPos`	Indices of the gamma components.
`sdNormNullInit`	Initial standard deviations of the normal components.
`rateExpNegInit`	Initial rates of the mirrored exponential components. Only relevant if mirrored exponential components are specified.
`rateExpPosInit`	Initial rates of the exponential components. Only relevant if exponential components are specified.
`shapeGamNegInit`	Initial shape parameters of the mirrored gamma components. Only relevant if mirrored gamma components are specified.
`scaleGamNegInit`	Initial scale parameters of the mirrored gamma components. Only relevant if mirrored gamma components are specified.
`shapeGamPosInit`	Initial shape parameters of the gamma components. Only relevant if gamma components are specified.
`scaleGamPosInit`	Initial scale parameters of the gamma components. Only relevant if gamma components are specified.
`piInit`	Initial weights of the components. If missing, all k components are assigned the same initial weight 1/k.
`classificationsInit`	Initial classifications of the data points. If missing, all data points are assigned to class floor(k/2) with k = number of components.
`dirichletParInit`	Initial concentration parameter of prior distribution assigned to the mixture weights.
`shapeDir`	Prior shape parameter of gamma distribution for concentration parameter of prior distribution assigned to the mixture weights.
`scaleDir`	Prior scale parameter of gamma distribution for concentration parameter of prior distribution assigned to the mixture weights.
`weightsPrior`	Prior distribution assigned to mixture weights. Available are the Finite-dimensional Dirichlet prior (”FDD”), also known as Dirichlet-multinomial process, and the Truncated Dirichlet process (”TDP”). Both are approximations to the Dirichlet process

	for a large number of components, while the Finite-dimensional Dirichlet prior is also suited for a small number of components as a special case of the Dirichlet distribution.
`sdAlpha`	Standard deviation of proposal distribution for concentration parameter of the prior distribution assigned to the mixture weights in the Metropolis-Hastings step incorporated in the Gibbs sampler. Only relevant if `weightsPrior="FDD"`.
`shapeNorm0`	Prior shape parameter of gamma distribution for precision of normal components.
`scaleNorm0`	Prior scale parameter of gamma distribution for precision of normal components.
`shapeExpNeg0`	Prior shape parameter of gamma distribution for parameter of mirrored exponential components. Only relevant if mirrored exponential components are specified.
`scaleExpNeg0`	Prior scale parameter of gamma distribution for parameter of mirrored exponential components. Only relevant if mirrored exponential components are specified.
`shapeExpPos0`	Prior shape parameter of gamma distribution for parameter of exponential components. Only relevant if exponential components are specified.
`scaleExpPos0`	Prior scale parameter of gamma distribution for parameter of exponential components. Only relevant if exponential components are specified.
`shapeGamNegAlpha0`	Prior shape parameter of gamma distribution for shape parameter of mirrored gamma components. Only relevant if mirrored gamma components are specified.
`shapeGamNegBeta0`	Prior scale parameter of gamma distribution for shape parameter of mirrored gamma components. Only relevant if mirrored gamma components are specified.
`scaleGamNegAlpha0`	Prior shape parameter of gamma distribution for scale parameter of mirrored gamma components. Only relevant if mirrored gamma components are specified.

`scaleGamNegBeta0` Prior scale parameter of gamma distribution for scale parameter of mirrored gamma components. Only relevant if mirrored gamma components are specified.

`shapeGamPosAlpha0` Prior shape parameter of gamma distribution for shape parameter of gamma components. Only relevant if gamma components are specified.

`shapeGamPosBeta0` Prior scale parameter of gamma distribution for shape parameter of gamma components. Only relevant if gamma components are specified.

`scaleGamPosAlpha0` Prior shape parameter of gamma distribution for scale parameter of gamma components. Only relevant if gamma components are specified.

`scaleGamPosBeta0` Prior scale parameter of gamma distribution for scale parameter of gamma components. Only relevant if gamma components are specified.

`itb` Number of iterations used for burn-in.

`nmc` Number of iterations after burn-in used for analysis.

`thin` Thinning value for the iterations after burn-in.

`average` Way of averaging across the posterior distribution to obtain estimates of model parameters.

Either `average="mean"` or `average="median"`. Note: For the allocation to components, results are given for posterior mean, median and maximum density regardless of the specification.

`sdShape` Standard deviation of proposal distribution for shape parameter of gamma components in the Metropolis-Hastings step incorporated in the Gibbs sampler. Only relevant if gamma components are specified.

Details

The convergence of Markov chains must be assessed prior to an interpretation of results. Inspection of trace plots via `plotChains` is therefore urgently recommended. Iterations for which one of the chains has not yet reached stationarity should not be taken into account for analysis and can be excluded by setting an appropriate burn-in value `itb`.

Autocorrelation between subsequent chain values can be reduced by thinning the chain, setting an appropriate value for `thin`. The number of iterations for the chains after the burn-in should be increased when the thinning is increased, to ensure that a sufficient number of iterations, as specified by `nmc`, is used for analysis. For the histone modification ChIP-seq and gene expression microarray example data, we used `itb=2000`, `nmc=100000` and `thin=10` in the final analyses.

Value

An object of class `MixModelBayes-class` storing results, data, priors, initial values and information about convergence.

Author(s)

Martin Schaefer (martin.schaefer@udo.edu)

See Also

`plotChains`, `MixModelBayes-class`

Examples

```
set.seed(1000)
z <- c(rnorm(1000, 0, 0.5), rnorm(1000, 0, 1))
mm <- bayesMixModel(z, normNull=1:2, sdNormNullInit=c(0.1, 0.2),
  piInit=c(1/2, 1/2), shapeNorm0=c(1, 1), scaleNorm0=c(1, 1),
  shapeExpNeg0=c(), scaleExpNeg0=c(),
  shapeExpPos0=c(), scaleExpPos0=c(), itb=100, nmc=500, thin=10)
mm
plotComponents(mm)
plotChains(mm, chain="pi")
z <- c(rnorm(200, 0, 1), rnorm(200, 0, 5), rexp(200, 0.1), -rexp(200, 0.1))
mm <- bayesMixModel(z, normNull=1:2, gamNeg=3, gamPos=4,
  sdNormNullInit=c(1, 1),
  shapeGamNegInit=1, scaleGamNegInit=1, shapeGamPosInit=1, scaleGamPosInit=1,
  shapeNorm0=c(1,3), scaleNorm0=c(1,3),
  shapeGamNegAlpha0=1, shapeGamNegBeta0=1,
  scaleGamNegAlpha0=1, scaleGamNegBeta0=1,
```

```
  shapeGamPosAlpha0=1, shapeGamPosBeta0=1,
  scaleGamPosAlpha0=1, scaleGamPosBeta0=1,
  itb=10, nmc=50, thin=1)
mm
plotComponents(mm)
plotChains(mm, chain="pi")
```

`plotChains` *Produces trace plots for a Bayesian mixture model*

Description

This function method draws trace plots for a Bayesian mixture model, e.g. visualizes the course of the Markov Chains. Inspection of the Markov Chains is important to determine convergence of the chains, which is necessary for sensible results.

Usage

```
plotChains(object, chain, component, itb = 1, thin = 1, cols, ...)
```

Arguments

`object`	An object of `MixModelBayes-class`
`chain`	A character of length one giving the name of the parameter whose chain should be plotted. Can be omitted if `component` is given. Then, all parameters of the given components are plotted.
`component`	An integer specifying the components whose parameter chains should be plotted. Can be omitted if `chain` is given. Then, all trace plots are generated for all components having the parameter specified via argument `chain`.
`itb`	Number of iterations used for burn-in. The burn-in is relative to the output of `bayesMixModel`, e.g., any burn-in specified here is added to the burn-in that was specified when calling `bayesMixModel`.
`thin`	Thinning value for the iterations after burn-in. The thinning is relative to the output of `bayesMixModel`, e.g., any thinning specified here multiplies by the thinning that was specified in `bayesMixModel`.

`cols` Number of columns to be used in the plot. Optional, if omitted, the number of columns and rows are choosen be the method itself.

`...` Further arguments passed to `plot`.

Details

The number of iterations necessary until a Markov chain reaches stationarity depends on the specific model and data. For any inference based on Markov Chain Monte Carlo methods, it is therefore necessary to inspect the convergence of Markov chains. One way to do this is visual inspection of trace plots using this method.

If argument `main` is passed to this method, it should have as many elements as chains are plotted. Otherwise, the vector `main` is repeated.

Author(s)

Hans-Ulrich Klein (h.klein@uni-muenster.de)

Martin Schaefer (martin.schaefer@udo.edu)

See Also

`bayesMixModel`, `MixModelBayes-class`

Examples

```
z <- c(rnorm(1000, 0, 3), rnorm(1000, 0, 5), rexp(1000, 5), -rexp(1000, 5))
mm <- bayesMixModel(z, normNull=1:2, expNeg=3, expPos=4,
  sdNormNullInit=c(1, 2), rateExpNegInit=8, rateExpPosInit=8,
  shapeNorm0=c(1, 1), scaleNorm0=c(1, 1),
  shapeExpNeg0=c(1, 1), scaleExpNeg0=c(1, 1),
  shapeExpPos0=c(1, 1), scaleExpPos0=c(1, 1),
  itb=200, nmc=1000, thin=10)
plotChains(mm, chain="pi")
plotChains(mm, component=c(2,3))
```

List of Abbreviations

AIC	**A**kaike **I**nformation **C**riterion
ALL	**A**cute **L**ymphoblastic **L**eukemia
AML	**A**cute **M**yeloid **L**eukemia
ATRA	**A**ll-**T**rans-**R**etinoic **A**cid
BIC	**B**ayesian **I**nformation **C**riterion
ChIP	**Ch**romation **I**mmuno**p**recipitation
CN	**C**opy **N**umber
DIC	**D**eviance **I**nformation **C**riterion
DNA	**D**eoxyribo**n**ucleic **A**cid
DPM	**D**irichlet **p**rocess **m**ixture
e.g.	**e**xampli **g**ratia (lat.), for instance
EM	**E**xpectation-**M**aximization
FPKM	**F**ragments **p**er **K**ilobase of Transcript per **M**illion Fragments Mapped
GE	**G**ene **E**xpression
HMM	**H**idden **M**arkov **M**odel
i.e.	**i**d **e**st (lat.), this means

LNCaP Human prostate adenocarcinoma cell line

MCMC **M**arkov **C**hain **M**onte **C**arlo

mRNA **m**essenger **R**ibo**n**ucleid **A**cid

PALM **P**hoto**a**ctivated **L**ocalization **M**icroscopy

PCR **P**olymerase **C**hain **R**eaction

PPM **P**roduct **P**artition **M**odel

PrEC **Pr**ostate **E**pithelial **C**ell

RNA **R**ibo**n**ucleid **A**cid

ROC **R**eceiver **O**perating **C**haracteristic

ROI **R**egion **O**f **I**nterest

SNP **S**ingle **N**ucleotide **P**olymorphism

SSM **S**pecies **S**ampling **M**odels

STORM **S**tochastic **O**ptical **R**econstruction **M**icroscopy

TCP **T**rans-2-Phenyl**c**yclo-**P**ropylamine

TIRF **T**otal **I**nternal **R**eflection **F**luorescence

TSS **T**ranscriptional **S**tart **S**ite

w.r.t **w**ith **r**espect **t**o

List of Figures

List of Tables

List of Algorithms

Bibliography

Akaike, H. (1974). A new look at statistical model identfication. *IEEE Transactions on Automatic Control*, **19**, 716–723.

Alberts, B., Bray, D., Hopkin, K., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P. (2010). *Essential Cell Biology*. Garland Science, third edition.

Anders, S. and Huber, W. (2010). Differential expression analysis for sequence count data. *Genome Biology*, **11**(10), R106.

Ando, T. (2007). Bayesian predictive information criterion for the evaluation of hierarchical Bayesian and empirical Bayes models. *Biometrika*, **94**(2), 443–458.

Angelini, C. and Costa, V. (2014). Understanding gene regulatory mechanisms by integrating ChIP-seq and RNA-seq data: statistical solutions to biological problems. *Frontiers in cell and developmental biology*, **2**, 2:51.

Ankerst, M., Breunig, M. M., Kriegel, H.-P., and Sander, J. (1999). OPTICS: Ordering points to identify the clustering structure. In *ACM SIGMOD International Conference on Management of Data,*, page 49–60. ACM Press.

Ansorge, W. J. (2009). Next-generation DNA sequencing techniques. *New Biotechnology*, **25**(4), 195–203.

Argiento, R., Cremaschi, A., and Guglielmi, A. (2013). A 'density-based' algorithm for cluster analysis using species sampling Gaussian mixture models. *Journal of Computational and Graphical Statistics*, **23**(4), 1126–1142.

Ashoor, H., Hérault, A., Kamoun, A., Radvanyi, F., Bajic, V. B., Barillot, E., and Boeva, V. (2013). HMCan: a method for detecting chromatin modifications in cancer samples using ChIP-seq data. *Bioinformatics*, **29**(23), 2979–2986.

Aspland, S. E., Bendall, H. H., and Murre, C. (2001). The role of E2A-PBX1 in leukemogenesis. *Oncogene*, **20**, 5708–5717.

Auer, P. L. and Doerge, R. W. (2011). A two-stage Poisson model for testing RNA-seq data. *Statistical applications in genetics and molecular biology*, **10**(1), 1–26.

Baddeley, A. and Turner, R. (2005). spatstat: An R package for analyzing spatial point patterns. *Journal of Statistical Software*, **12**(6), 1–42.

Baddeley, A., Møller, J., and Waagepetersen, R. (2000). Non- and semiparametric estimation of interaction in inhomogeneous point patterns. *Statistica Neerlandica*, **54**, 329–350.

Baddeley, A. J. and Møller, J. (1989). Nearest-neighbour Markov point processes and random sets. *International Statistical Review*, **57**, 89–121.

Baddeley, A. J. and Silverman, B. W. (1984). A cautionary example on the use of 2nd-order methods for analyzing point patterns. *Biometrics*, **40**(4), 1089–1093.

Baddeley, A. J. and van Lieshout, M. N. M. (1995). Area-interaction point processes. *Annals of the Institute of Statistical Mathematics*, **47**, 601–619.

Banfield, J. D. and Raftery, A. E. (1993). Model-based Gaussian and non-Gaussian clustering. *Biometrics*, **49**, 803–821.

Bates, M., Huang, B., Dempsey, G. T., and Zhuang, X. (2007). Multicolor super-resolution imaging with photo-switchable fluorescent probes. *Science*, **317**, 1749–1753.

Baudry, J. P., Raftery, A., Celeux, G., Lo, K., and Gottardo, R. (2010). Combining mixture components for clustering. *Journal of Computational and Graphical Statistics*, **19**(2), 332–353.

Baylin, S. and Jones, P. (2011). A decade of exploring the cancer epigenome - biological and translational implications. *Nat Rev Cancer*, **11**(10), 726–734.

Beal, M. J. and Ghahramani, Z. (2003). The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. In

J. Bernardo, M. Bayarri, J.O.Berger, A.P.Dawid, D. Heckerman, A. Smith, and M. West, editors, *Bayesian Statistics 7*. Oxford University Press, Oxford.

Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society B*, **57**(1), 289–300.

Bert, S. A., Robinson, M. D., Strbenac, D., Statham, A. L., Song, J. Z., Hulf, T., Sutherland, R. L., Coolen, M. W., Stirzaker, C., and Clark, S. J. (2013). Regional activation of the cancer genome by long-range epigenetic remodeling. *Cancer Cell*, **23**(1), 9–22.

Besag, J. and Diggle, P. J. (1977). Simple Monte Carlo tests for spatial pattern. *Journal of the Royal Statistical Society Series C*, **26**(3), 327–333.

Besag, J., York, J., and Mollie, A. (1991). Bayesian image restoration, with two applications in spatial statistics. *Annals of the Institute of Statistical Mathematics*, **43**, 1–59.

Betzig, E., Patterson, G. H., Sougrat, R., Lindwasser, O. W., Olenych, S., Bonifacino, J. S., Davidson, M. W., Lippincott-Schwartz, J., and Hess, H. (2006). Imaging intracellular fluorescent proteins at nanometer resolution. *Science*, **313**, 1642–1645.

Biankin, A. V., Waddell, N., Kassahn, K. S., Gingras, M.-C., Muthuswamy, L. B., Johns, A. L., Miller, D. K., Wilson, P. J., Patch, A.-M., Wu, J., Chang, D. K., Cowley, M. J., Gardiner, B. B., Song, S., Harliwong, I., Idrisoglu, S., Nourse, C., Nourbakhsh, E., Manning, S., Wani, S., Gongora, M., Pajic, M., Scarlett, C. J., Gill, A. J., Pinho, A. V., Rooman, I., Anderson, M., Holmes, O., Leonard, C., Taylor, D., Wood, S., Xu, Q., Nones, K., Fink, J. L., Christ, A., Bruxner, T., Cloonan, N., Kolle, G., Newell, F., Pinese, M., Mead, R. S., Humphris, J. L., Kaplan, W., Jones, M. D., Colvin, E. K., Nagrial, A. M., Humphrey, E. S., Chou, A., Chin, V. T., Chantrill, L. A., Mawson, A., Samra, J. S., Kench, J. G., Lovell, J. A., Daly, R. J., Merrett, N. D., Toon, C., Epari, K., Nguyen, N. Q., Barbour, A., Zeps, N., Initiative, A. P. C. G., Kakkar, N., Zhao, F., Wu, Y. Q., Wang, M., Muzny, D. M., Fisher, W. E., Brunicardi, F. C., Hodges, S. E., Reid, J. G.,

Drummond, J., Chang, K., Han, Y., Lewis, L. R., Dinh, H., Buhay, C. J., Beck, T., Timms, L., Sam, M., Begley, K., Brown, A., Pai, D., Panchal, A., Buchner, N., Borja, R. D., Denroche, R. E., Yung, C. K., Serra, S., Onetto, N., Mukhopadhyay, D., Tsao, M.-S., Shaw, P. A., Petersen, G. M., Gallinger, S., Hruban, R. H., Maitra, A., Iacobuzio-Donahue, C. A., Schulick, R. D., Wolfgang, C. L., Morgan, R. A., Lawlor, R. T., Capelli, P., Corbo, V., Scardoni, M., Tortora, G., Tempero, M. A., Mann, K. M., Jenkins, N. A., Perez-Mancera, P. A., Adams, D. J., Largaespada, D. A., Wessels, L. F. A., Rust, A. G., Stein, L. D., Tuveson, D. A., Copeland, N. G., Musgrove, E. A., Scarpa, A., Eshleman, J. R., Hudson, T. J., Sutherland, R. L., Wheeler, D. A., Pearson, J. V., McPherson, J. D., Gibbs, R. A., and Grimmond, S. M. (2012). Pancreatic cancer genomes reveal aberrations in axon guidance pathway genes. *Nature*, **491**(7424), 399–405.

Binder, D. A. (1978). Bayesian cluster analysis. *Biometrika*, **65**, 31–38.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet allocation. *J. Mach. Learn. Res.*, **3**, 993–1022.

Blei, D. M., Griffiths, T. L., and Jordan, M. I. (2010). The nested chinese restaurant process and Bayesian nonparametric inference of topic hierarchies. *Journal of the ACM*, **57**(2).

Bolouri, H. (2014). Modeling genomic regulatory networks with big data. *Trends in Genetics*, **30**(5), 182–191.

Bolstad, B. M., Irizarry, R. A., Astrand, M., and Speed, T. P. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, **19**(2), 185–193.

Broet, P. and Richardson, S. (2006). Detection of gene copy number changes in CGH microarrays using a spatially correlated mixture model. *Bioinformatics*, **22**, 911–918.

Brown, K. (2013). Distances in bounded regions. MathPages website, URL: http://www.mathpages.com/home/kmath324/kmath324.htm (last access 20.05.2014).

Burnham, K. P. and Anderson, D. R. (2002). *Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach*. Springer, second edition.

Byers, S. and Raftery, A. E. (1998). Nearest-neighbour clutter removal for estimating features in spatial point processes. *Journal of the American Statistical Association*, **93**(442), 577–584.

Byth, K. and Ripley, B. D. (1980). On sampling spatial patterns by distance methods. *Biometrics*, **36**, 279–284.

Cha, M. and Zhou, Q. (2014). Detecting clustering and ordering binding patterns among transcription factors via point process models. *Bioinformatics*, **30**(16), 2263–2271.

Cheezum, M. K., Walker, W. F., and Guilford, W. H. (2001). Quantitative comparison of algorithms for tracking single fluorescent particles. *Biophysical Journal*, **81**, 2378–2388.

Chen, L. (2011). Statistical and computational studies on alternative splicing. In H. H.-S. Lu, B. Schölkopf, and H. Zhao, editors, *Handbook of Statistical Bioinformatics*, pages 31–53. Springer.

Chen, Y., Meyer, C. A., Liu, T., Li, W., Liu, J. S., and Liu, X. S. (2011). MM-ChIP enables integrative analysis of cross-platform and between-laboratory ChIP-chip or ChIP-seq data. *Genome Biology*, **12**, R11.

Cheng, C., Yan, K.-K., Yip, K. Y., Rozowsky, J., Alexander, R., Shou, C., and Gerstein, M. (2011). A statistical framework for modeling gene expression using chromatin features and application to modENCODE datasets. *Genome Biology*, **12**(2), R15.

Cheng, C., Alexander, R., Min, R., Leng, J., Yip, K. Y., Rozowsky, J., Yan, K.-K., Dong, X., Djebali, S., Ruan, Y., Davis, C. A., Carninci, P., Lassman, T., Gingeras, T. R., Guigó, R., Birney, E., Weng, Z., Snyder, M., and Gerstein, M. (2012). Understanding transcriptional regulation by integrative analysis of transcription factor binding data. *Genome Research*, **22**(9), 1658–1667.

Choi, H., Nesvizhskii, A. I., Ghosh, D., and Qin, Z. S. (2009). Hierarchical hidden Markov model with application to joint analysis of ChIP-chip and ChIP-seq data. *Bioinformatics*, **25**(14), 1715–1721.

Chung, L., Ferguson, J., Zheng, W., Qian, F., Bruno, V., Montgomery, R., and Zhao, H. (2013). Differential expression analysis for paired RNA-seq data. *BMC Bioinformatics*, **14**, 110.

Citri, A. and Yarden, Y. (2006). EGF-erbB signalling: towards the systems level. *Nature Reviews Molecular Cell Biology*, **7**(7), 505–516.

Cox, D. R. (1955). Some statistical methods connected with series of events. *Journal of the Royal Statistical Society B*, **17**, 129–164.

Cressie, N. (1993). *Statistics for spatial data*. Wiley.

Crowley, E. M. (1997). Product partition models for normal means. *Journal of the American Statistical Association*, **92**(437), 192–198.

Dahl, D. B. (2006). Model-based clustering for expression data via a Dirichlet process mixture model. In K. A. Do, P. Müller, and M. Vannucci, editors, *Bayesian inference for gene expression and proteomics*, pages 201–218. Cambridge University Press.

Daley, D. and Vere-Jones, D. (2008). *An introduction to the theory of point processes*. Springer, second edition.

Dasgupta, A. and Raftery, A. E. (1998). Detecting features in spatial point processes with clutter via model-based clustering. *Journal of the American Statistical Association*, **93**(441), 294–302.

Dass, S. C., Lim, C. Y., Maiti, T., and Zhang, Z. (2014). Clustering curves based on change point analysis: A nonparametric Bayesian approach. *Statistica Sinica*, **25**, 677–708.

Dawson, M. A. and Kouzarides, T. (2012). Cancer epigenetics: from mechanism to therapy. *Cell*, **150**(1), 12–27.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society B*, **39**, 1–38.

Dennis, G., Sherman, B. T., Hosack, D. A., Yang, J., Gao, W., Lane, H. C., and Lempicki, R. A. (2003). David: Database for annotation, visualization, and integrated discovery. *Genome Biology*, **4**(5), P3.

Di, Y., Schafer, D. W., Cumbie, J. S., and Chang, J. H. (2011). The NBP negative binomial model for assessing differential gene expression from RNA-seq. *Statistical Applications in Genetics and Molecular Biology*, **10**(1), 1–28.

Diggle, P. (2003). *Statistical analysis of spatial point patterns*. Arnold, second edition.

Dixon, P. M. (2002). Ripley's K function. In A. H. El-Shaarawi and W. W. Piegorsch, editors, *Encyclopedia of Environmetrics*, volume 3, page 1796–1803. John Wiley and Sons.

Do, K.-A., Müller, P., and Tang, F. (2005). A Bayesian mixture model for differential gene expression. *Applied Statistics*, **54**, 627–644.

Dobin, A., Davis, C. A., Schlesinger, F., Drenkow, J., Zaleski, C., Jha, S., Batut, P., Chaisson, M., and Gingeras, T. R. (2013). STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, **29**(1), 15–21.

Dong, X., Greven, M. C., Kundaje, A., Djebali, S., Brown, J. B., Cheng, C., Gingeras, T. R., Gerstein, M., Guigó, R., Birney, E., and Weng, Z. (2012). Modeling gene expression using chromatin features in various cellular contexts. *Genome Biol*, **13**(9), R53.

Dunson, D. B. (2010). Nonparametric Bayes applications to biostatistics. In N. L. Hjort, C. Holmes, P. Müller, and S. Walker, editors, *Bayesian Nonparametrics*, pages 223–273. Cambridge University Press, Cambridge.

Elling, C., Erben, P., Walz, C., Frickenhaus, M., Schemionek, M., Stehling, M., Serve, H., Cross, N. C. P., Hochhaus, A., Hofmann, W.-K., Berdel, W. E., Müller-Tidow,

C., Reiter, A., and Koschmieder, S. (2011). Novel imatinib-sensitive PDGFRA-activating point mutations in hypereosinophilic syndrome induce growth factor independence and leukemia-like disease. *Blood*, **117**, 2935–2943.

Emmert-Streib, F., V.Glazko, G., Altay, G., and de Matos Simoes, R. (2012). Statistical inference and reverse engineering of gene regulatory networks from observational expression data. *Frontiers in Genetics*, **3**, 8.

Ernst, J. and 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. and West, M. (1995). Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, **90**, 577–588.

Esnaola, M., Puig, P., Gonzalez, D., Castelo, R., and Gonzalez, J. R. (2013). A flexible count data model to fit the wide diversity of expression profiles arising from extensively replicated RNA-seq experiments. *BMC Bioinformatics*, **14**, 254.

Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In E. Simoudis, J. Han, and U. Fayyad, editors, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96)*, page 226–231. AAAI Press.

Everitt, B. and Hand, D. (1981). *Finite mixture distributions*. Chapman and Hall, London.

Fahrmeir, L. and Lang, S. (2000). Bayesian semiparametric regression analysis of multicategorical time-space data. *Annals of the Institute of Statistical Mathematics*, **53**, 10–30.

Fasan, A., Haferlach, C., Alpermann, T., Jeromin, S., Grossmann, V., Eder, C., Weissmann, S., Dicker, F., Kohlmann, A., Schindela, S., Kern, W., Haferlach, T., and Schnittger, S. (2014). The role of different genetic subtypes of CEBPA mutated AML. *Leukemia*, **28**, 794–803.

Fejes, A., Robertson, G., Bilenky, M., Varhol, R., Bainbridge, M., and Jones, S. (2008). Findpeaks 3.1: a tool for identifying areas of enrichment from massively parallel short-read sequencing technology. *Bioinformatics*, **24**(15), 1729–1730.

Ferguson, T. S. (1973). A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, **1**, 209–230.

Ferguson, T. S. (1983). Bayesian density estimation by mixtures of normal distributions. In M. Rizvi, J. Rustagi, and D. Siegmund, editors, *Recent advances in Statistics*, pages 287–302. Academic Press, New York.

Fernández-Medarde, A. and Santos, E. (2011). Ras in cancer and developmental diseases. *Genes & Cancer*, **2**(3), 344–358.

Flicek, P., Ahmed, I., Amode, M. R., Barrell, D., Beal, K., Brent, S., Carvalho-Silva, D., Clapham, P., Coates, G., Fairley, S., Fitzgerald, S., Gil, L., Garcia-Girón, C., Gordon, L., Hourlier, T., Hunt, S., Juettemann, T., Kähäri, A., Keenan, S., Komorowska, M., Kulesha, E., Longden, I., Maurel, T., McLaren, W., Muffato, M., Nag, R., Overduin, B., Pignatelli, M., Pritchard, B., Pritchard, E., Riat, H. S., Ritchie, G. R. S., Ruffier, M., Schuster, M., Sheppard, D., Sobral, D., Taylor, K., Thormann, A., Trevanion, S., White, S., Wilder, S. P., Aken, B. L., Birney, E., Cunningham, F., Dunham, I., Harrow, J., Herrero, J., Hubbard, T. J. P., Johnson, N., Kinsella, R., Parker, A., Spudich, G., Yates, A., Zadissa, A., and Searle, S. M. J. (2013). Ensembl 2013. *Nucleic Acids Research*, **41**, D48–D55.

Fraley, C. and Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, **97**, 611–631.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *Annals of Statistics*, **19**(1), 1–67.

Fritsch, A. (2010). *Bayesian mixtures for cluster analysis and flexible modeling of distributions*. Ph.D. thesis, Department of Statistics, TU Dortmund University.

Fritsch, A. and Ickstadt, K. (2009). Improved criteria for clustering based on the posterior similarity matrix. *Bayesian Analysis*, **4**, 367–392.

Frühwirth-Schnatter, S. (2006). *Finite mixture and Markov switching models.* Springer, Berlin.

Frühwirth-Schnatter, S. (2011). Dealing with label switching under model uncertainty. In K. Mengersen, C. Robert, and D. Titterington, editors, *Mixture estimation and applications*, pages 193–218. Wiley, Chichester.

Furey, T. S. (2012). ChIP-seq and beyond: new and improved methodologies to detect and characterize protein-DNA interactions. *Nature Reviews Genetics*, **13**(12), 840–852.

Gelfand, A. E. and Smith, A. F. M. (1990). Sampling-based approaches for calculating marginal densities. *Journal of the American Statistical Association*, **85**, 398–409.

Gelfand, A. E., Guindani, M., and Petrone, S. (2007). Bayesian nonparametric modelling for spatial data using Dirichlet processes. In J. Bernardo, J. Bayarri, J. Berger, A. Dawid, D. Heckerman, A. Smith, and M. West, editors, *Bayesian Statistics 8*, pages 1–26. Oxford University Press, Oxford.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian data analysis*. Chapman and Hall.

Ghosh, D. and Qin, Z. S. (2010). Statistical issues in the analysis of ChIP-Seq and RNA-seq data. *Genes*, **1**, 317–334.

Ghosh, J. and Ramamoorthi, R. (2003). *Bayesian Nonparametrics*. Springer.

Giorgi, F. M., Fabbro, C. D., and Licausi, F. (2013). Comparative study of RNA-seq- and microarray-derived coexpression networks in arabidopsis thaliana. *Bioinformatics*, **29**(6), 717–724.

Gong, W., Koyano-Nakagawa, N., Li, T., and Garry, D. J. (2015). Inferring dynamic gene regulatory networks in cardiac differentiation through the integration of multi-dimensional data. *BMC Bioinformatics*, **16**, 74.

Goreaud, F. and Pélissier, R. (1999). On explicit formulas of edge effect correction for Ripley's K-function. *Journal of Vegetation Science*, **10**(3), 433–438.

Grabarnik, P. and Särkka, A. (2001). Interacting neighbour point processes: Some models for clustering. *Journal of Statistical Computation and Simulation*, **68**, 103–125.

Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, **82**, 711–732.

Green, P. J. and Richardson, S. (2001). Modelling heterogeneity with and without the Dirichlet process. *Scandinavian Journal of Statistics*, **28**, 355–376.

Griffin, J. E. and Steel, M. F. J. (2006). Order-based dependent Dirichlet processes. *Journal of the American Statistical Association*, **101**, 179–194.

Guan, D., Shao, J., Deng, Y., Wang, P., Zhao, Z., Liang, Y., J., W., and Yan, B. (2014a). CMGRN: a webserver for constructing multilevel gene regulatory networks using ChIP-seq and gene expression data. *Bioinformatics*, **30**, 1190–1192.

Guan, D., Shao, J., Zhao, Z., Wang, P., Qin, J., Deng, Y., Boheler, K. R., Wang, J., and Yan, B. (2014b). PTHGRN: unraveling post-translational hierarchical gene regulatory networks using PPI, ChIP-seq and gene expression data. *Nucleic Acids Research*, **42**, W130–W136.

Guan, Y., Myers, C. L., Lu, R., Lemischka, I. R., Bult, C. J., and Troyanskaya, O. G. (2008). A genomwide functional network for the laboratory mouse. *PloS Computational Biology*, **4**(9), e1000165.

Guindani, M., Sepúlveda, N., Paulino, C. D., and Müller, P. (2014). A Bayesian semi-parametric approach for the differential analysis of sequence counts data. *Journal of the Royal Statistical Society C*, **63**(3), 385–404.

Gurry, T., Kahramanogullari, O., and Endres, R. G. (2009). Biophysical mechanism for Ras-nanocluster formation and signaling in plasma membrane. *PLoS ONE*, **4**(7), e6148.

Hancock, J. F. (2006). Lipid rafts: Contentious only from simplistic standpoints. *Nature Reviews Molecular Cell Biology*, **7**(6), 456–462.

Hardcastle, T. J. and Kelly, K. (2010). baySeq: empirical Bayesian methods for identifying differential expression in sequence count data. *BMC Bioinformatics*, **11**, 422–422.

Harrow, J., Frankish, A., Gonzalez, J. M., Tapanari, E., Diekhans, M., Kokocinski, F., Aken, B. L., Barrell, D., Zadissa, A., Searle, S., Barnes, I., Bignell, A., Boychenko, V., Hunt, T., Kay, M., Mukherjee, G., Rajan, J., Despacio-Reyes, G., Saunders, G., Steward, C., Harte, R., Lin, M., Howald, C., Tanzer, A., Derrien, T., Chrast, J., Walters, N., Balasubramanian, S., Pei, B., Tress, M., Rodriguez, J. M., Ezkurdia, I., van Baren, J., Brent, M., Haussler, D., Kellis, M., Valencia, A., Reymond, A., Gerstein, M., Guigó, R., and Hubbard, T. J. (2012). GENCODE: the reference human genome annotation for The ENCODE Project. *Genome Research*, **22**(9), 1760–1774.

Hartigan, J. A. (1990). Partition models. *Communications in Statistics, Part A – Theory and Methods*, **19**, 2745–56.

Hasemann, M. S., Lauridsen, F. K. B., Waage, J., Jakobsen, J. S., Frank, A.-K., Schuster, M. B., Rapin, N., Bagger, F. O., Hoppe, P. S., Schroeder, T., and Porse, B. T. (2014). C/ebp α is required for long-term self-renewal and lineage priming of hematopoietic stem cells and for the maintenance of epigenetic configurations in multipotent progenitors. *PLoS Genetics*, **10**(1), e1004079.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, **57**(1), 97–109.

Hawkins, R. D., Hon, G. C., and Ren, B. (2010). Next-generation genomics: an integrative approach. *Nature Reviews Genetics*, **11**(7), 476–486.

Hebenstreit, D., Gu, M., Haider, S., Turner, D. J., Liò, P., and Teichmann, S. A. (2011). EpiChIP: gene-by-gene quantification of epigenetic modification levels. *Nucleic Acids Research*, **39**(5), e27.

Hebestreit, K. (2009). *Detektion und räumliche Analyse von Ras-Proteinen auf der Zellmembran*. Diploma thesis, Faculty of Statistics, TU Dortmund University.

Henis, Y. I., Hancock, J. F., and Prior, I. A. (2009). Ras acylation, compartmentalization and signaling nanoclusters (review). *Molecular Membrane Biology*, **26**, 80–92.

Hennig, C. (2014). *fpc: Flexible procedures for clustering*, R package version 2.1.9 edition.

Hennig, C. and Coretto, P. (2008). The noise component in model-based cluster analysis. In C. Preisach, H. Burkhardt, L. Schmidt-Thieme, and R. Decker, editors, *Data Analysis, Machine Learning and Applications*, pages 127–138. Springer, Berlin.

Herrmann, S., Schwender, H., Ickstadt, K., and Müller, P. (2014). A Bayesian changepoint analysis of ChIP-seq data of Lamin B. *BBA*, **1844**(1), 138–144.

Herrmann, S., Ickstadt, K., Schäfer, M., Radon, Y., and Verveer, P. (2015). Approach for analysing dual colour data. *unsubmitted manuscript*.

Hess, S., Girirajan, T. P. K., and Masony, M. D. (2006). Ultra-high resolution imaging by fluorescence photoactivation localization microscopy. *Biophysical Journal*, **91**(11), 4258–4272.

Hjort, N. L., Holmes, C., Müller, P., and Walker, S., editors (2010). *Bayesian Nonparametrics*. Cambridge University Press, Cambridge.

Ho, W. K. J., Bishop, E., Karchenko, P. V., Negré, N., White, K. P., and Park, P. J. (2011). ChIP-chip versus ChIP-seq. Lessons for experimental design and data analysis. *BMC Genomics*, **12**, 134.

Hoffman, M. M., Buske, O. J., Wang, J., Weng, Z., Bilmes, J. A., and Noble, W. S. (2012). Unsupervised pattern discovery in human chromatin structure through genomic segmentation. *Nature Methods*, **9**(5), 473–476.

Holmes, I., Harris, K., and Quince, C. (2012). Dirichlet multinomial mixtures: Generative models for microbial metagenomics. *PLoS ONE*, **7**(2), e30126.

Hower, V., Evans, S. N., and Pachter, L. (2011). Shape-based peak identification for ChIP-Seq. *BMC Bioinformatics*, **12**, 15.

Hu, M., Zhu, Y., Taylor, J. M., Liu, J., and Qui, Z. (2011). Using poisson mixed-effects model to quantify transcript-level gene expression in RNA-Seq. *Bioinformatics*, **28**(1), 63–68.

Hung, R., Yu, G., Wang, Z., Zhang, J., and Shi, L. (2013). Dirichlet process mixture model for document clustering with feature partition. *IEEE Transactions on Knowledge and Data Engineering*, **25**(8), 1748–1759.

Hurvich, C. M. and Tsai, C.-L. (1989). Regression and time series model selection in small samples. *Biometrika*, **76**, 297–307.

Huss, M. (2010). Introduction into the analysis of high-throughput-sequencing based epigenome data. *Briefings in Bioinformatics*, **2**(5), 512–523.

Ibrahim, M. M., Lacadie, S. A., and Ohle, U. (2015). JAMM: a peak finder for joint analysis of NGS replicates. *Bioinformatics*, **31**(1), 48–55.

Illian, J., Panttinen, A., Stoyan, H., and Stoyan, D. (2008). *Statistical analysis and modelling of spatial point patterns*. Wiley.

Irizarry, R. A., Hobbs, B., Collin, F., Beazer-Barclay, Y. D., Antonellis, K. J., Scherf, U., and Speed, T. P. (2003). Exploration, normalization, and summaries of high density oligonucleotide array probe level data. *Biostatistics*, **4**(2), 249–264.

Ishwaran, H. and James, L. F. (2001). Gibbs sampling methods for stick-breaking priors. *Journal of the American Statistical Association*, **96**, 161–173.

Ishwaran, H. and James, L. F. (2003). Generalized weighted Chinese restaurant processes for species sampling mixture models. *Statistica Sinica*, **13**, 1211–1235.

Ishwaran, H. and Zarepour, M. (2000). Markov chain Monte Carlo in approximate Dirichlet and beta two-parameter process hierarchical models. *Biometrika*, **87**(2), 371–390.

Ishwaran, H. and Zarepour, M. (2002a). Dirichlet prior sieves in finite normal mixtures. *Statistica Sinica*, **12**, 941–963.

Ishwaran, H. and Zarepour, M. (2002b). Exact and approximate sum representations for the Dirichlet process. *The Canadian Journal of Statistics*, **30**(2), 269–283.

Jacobson, K., Mouritsen, O. G., and Anderson, R. G. (2007). Lipid rafts: At a crossroad between cell biology and physics. *Nature Cell Biology*, **9**, 7–14.

Jara, A., Hanson, T., Quintana, F., Müller, P., and Rosner, G. (2011). DPpackage: Bayesian semi- and nonparametric modeling in R. *Journal of Statistical Software*, **40**(5), 1–30.

Jasra, A., Holmes, C. C., and Stephens, D. A. (2005). Markov chain Monte Carlo methods and the label switching problem in Bayesian mixture modeling. *Statistical Science*, **20**, 50–67.

Ji, C., Merl, D., Kepler, T. B., and West, M. (2009). Spatial mixture modelling for unobserved point processes: Examples in immunofluorescence histology. *Bayesian Analysis*, **4**, 297–316.

Ji, H., Jiang, H., Ma, W., Johnson, D. S., Myers, R. M., and Wong, W. H. (2008). An integrated software system for analyzing ChIP-chip and ChIP-seq. *Nature Biotechnology*, **26**(11), 1293–1300.

Jiang, H. and Wong, W. H. (2009). Statistical inferences for isoform expression in RNA-seq. *Bioinformatics*, **25**, 1026–1032.

Johannes, F., Wardenaar, R., Colomé-Tatché, M., Mousson, F., de Graaf, P., Mokry, M., Guryev, V., Timmers, H. T. M., Cuppen, E., and Jansen, R. C. (2010). Comparing genome-wide chromatin profiles using ChIP-chip or ChIP-seq. *Bioinformatics*, **26**(8), 1000–1006.

Johnson, S. (1967). Hierarchical clustering schemes. *Psychometrika*, **32**, 241–254.

Joseph, J., Doshi-Velez, F., Huang, A. S., and Roy, N. (2011). A Bayesian nonparametric approach to modeling motion patterns. *Autonomous Robots*, **31**, 383–400.

Joshi, A., van de Peer, Y., and Michoel, T. (2008). Analysis of a Gibbs sampler method for model-based clustering of gene expression data. *Bioinformatics*, **24**(2), 176–183.

Kamps, M. P. and Baltimore, D. (1993). E2A-Pbx-1, the t(1;19) translocation protein of human pre-b-cell acute lymphocytic leukemia, causes acute myeloid leukemia in mice. *Molecular and Cell Biology*, **13**, 351–357.

Kandala, N.-B., Lang, S., Klasen, S., and Fahrmeir, L. (2001). Semiparametric analysis of the socio-demographic determinants of undernutrition in two African countries. *Research in Official Statistics*, **4**(1), 81–100.

Karlić, R., Chung, H., Lasserre, J., Vlahovicek, K., and Vingron, M. (2010). Histone modification levels are predictive for gene expression. *Proceedings of the National Academy of Sciences of the United States of America*, **107**, 2926–2931.

Karlis, K. and Meligkotsidou, L. (2007). Finite mixtures of multivariate poisson distributions with application. *Journal of Statistical Planning and Inference*, **137**, 1942–1960.

Kaufman, L. and Rousseeuw, P. (1990). *Finding groups in data: An introduction to cluster analysis.* Wiley, New York.

Kiel, M. J., Yilmaz, O. H., Iwashita, T., Yilmaz, O. H., Terhorst, C., and Morrison, S. J. (2005). SLAM family receptors distinguish hematopoietic stem and progenitor cells and reveal endothelial niches for stem cells. *Cell*, **121**(7), 1109–1121.

Kim, S., Tadesse, M. G., and Vannucci, M. (2006). Variable selection in clustering via Dirichlet process mixture models. *Biometrika*, **93**, 877–893.

Kirk, P., Griffin, J. E., Savage, R. S., Ghahramani, Z., and Wild, D. L. (2012). Bayesian correlated clustering to integrate multiple datasets. *Bioinformatics*, **28**, 3290–3297.

Kiskowski, M. A., Hancock, J. F., and Kenworthy, A. K. (2009). On the use of Ripley's K-function and its derivatives to analyze domain size. *Biophysical Journal*, **97**, 1095–1103.

Klambauer, G., Schwarzbauer, K., Mayr, A., Clevert, D.-A., Mitterecker, A., Bodenhofer, U., and Hochreiter, S. (2012). cn.MOPS: mixture of Poissons for discovering copy number variations in next-generation sequencing data with a low false discovery rate. *Nucleic Acids Research*, **40**(9), e69.

Klein, H.-U. and Schäfer, M. (2014). *epigenomix: Epigenetic and gene transcription data normalization and integration with mixture models*, R package version 1.7.2 edition.

Klein, H.-U., Schäfer, M., Porse, B., Hasemann, M., Ickstadt, K., and Dugas, M. (2014). Integrative analysis of histone ChIP-seq and gene expression microarray data using Bayesian mixture models. *Bioinformatics*, **30**(8), 1154–1162.

Kohonen, T. (1995). *Self-organizing maps*. Springer, Berlin.

Kormaksson, M., Booth, J. G., Figueroa, M. E., and Melnick, A. (2012). Integrative model-based clustering of microarray methylation and expression data. *The Annals of Applied Statistics*, **6**(3), 1327–1347.

Kornacker, K., Rye, M., Handstad, T., and Drablos, F. (2012). The Triform algorithm: improved sensitivity and specificity in ChIP-seq peak finding. *BMC Bioinformatics*, **13**(1), 176.

Kottas, A. and Sansó, B. (2007). Bayesian mixture modeling for spatial Poisson process intensities, with applications to extreme value analysis. *Journal of Statistical Planning and Inference*, **137**(10), 3151–3163. Special Issue: Bayesian Inference for Stochastic Processes.

Kriegel, H.-P., Kröger, P., Sander, J., and Zimek, A. (2011). Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **1**(3), 231–240.

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

Lahti, L., Schäfer, M., Klein, H.-U., Bicciato, S., and Dugas, M. (2013). Cancer gene prioritization by integrative analysis of mRNA expression and DNA copy number data: a comparative review. *Briefings in Bioinformatics*, **14**(1), 27–35.

Langmead, B., Hansen, K. D., and Leek, J. T. (2010). Cloud-scale RNA-sequencing differential expression analysis with myrna. *Genome Biology*, **11**(8), R83.

Lau, J. W. and Green, P. J. (2007). Bayesian model-based clustering procedures. *Journal of Computational and Graphical Statistics*, **16**, 526–558.

Law, C. W., Chen, Y., Shi, W., and Smyth, G. K. (2014). voom: precision weights unlock linear model analysis tools for RNA-seq read counts. *Genome Biology*, **15**, R29.

Lawson, A. and Denison, D., editors (2002). *Spatial cluster modeling*. Chapman and Hall.

Lee, H. J., Daver, N., Kantarjian, H. M., Verstovsek, S., and Ravandi, F. (2013). The role of JAK pathway dysregulation in the pathogenesis and treatment of acute myeloid leukemia. *Clinical Cancer Research*, **19**(2), 327–35.

Lee, I., Blom, U. M., Wang, P. I., Shim, J. E., and Marcotte, E. M. (2011). Prioritizing candidate disease genes by network-based boosting of genome-wide association data. *Genome Research*, **21**, 1109–1121.

Leng, N., Dawson, J. A., Thomson, J. A., Ruotti, V., Rissman, A. I., Smits, B. M. G., Haag, J. D., Gould, M. N., Stewart, R. M., and Kendziorski, C. (2013). EBSeq: an empirical Bayes hierarchical model for inference in RNA-seq experiments. *Bioinformatics*, **29**(8), 1035–1043.

Leng, N., Li, Y., McIntosh, B. E., Nguyen, B. K., Duffin, B., Tian, S., Thomson, J. A., Dewey, C. N., Stewart, R., and Kendziorski, C. (2015). EBSeq-HMM: a Bayesian approach for identifying gene-expression changes in ordered RNA-seq experiments. *Bioinformatics*, **31**(16), 2614–2622.

Li, H. and Durbin, R. (2009). Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics*, **25**(14), 1754–1760.

Li, J. and Tibshirani, R. (2013). Finding consistent patterns: a nonparametric approach for identifying differential expression in RNA-seq data. *Statistical Methods in Medical Research*, **22**(5), 519–536.

Li, J., Witten, D. M., Johnstone, I. M., and Tibshirani, R. (2012). Normalization, testing, and false discovery rate estimation for RNA-sequencing data. *Biostatistics*, **13**(3), 523–538.

Li, W., Dai, C., Kang, S., and Zhou, X. J. (2014). Integrative analysis of many RNA-seq datasets to study alternative splicing. *Methods*, **67**(1), 313–324.

Lijoi, A. and Prünster, I. (2010). Models beyond the Dirichlet process. In N. L. Hjort, C. Holmes, P. Müller, and S. Walker, editors, *Bayesian Nonparametrics*, pages 80–136. Cambridge University Press, Cambridge.

Lijoi, A., Nipoti, B., and Prünster, I. (2014). Dependent mixture models: Clustering and borrowing information. *Computational Statistics & Data Analysis*, **71**, 417–433.

Lin, S. M., Du, P., Huber, W., and Kibbe, W. A. (2008). Model-based variance-stabilizing transformation for Illumina microarray data. *Nucleic Acids Research*, **36**(2), e11.

Liu, J. S. (1996). Nonparametric hierarchical Bayes via sequential imputations. *The Annals of Statistics*, **24**, 911–930.

Liu, X., Sivaganesan, S., Yeung, K. Y., Guo, J., Bumgarner, R. E., and Medvedovic, M. (2006). Context-specific infinite mixtures for clustering gene expression profiles across diverse microarray dataset. *Bioinformatics*, **22**(14), 1737–1744.

Liu, X., Jessen, W. J., Sivaganesan, S., Aronow, B. J., and Medvedovic, M. (2007). Bayesian hierarchical model for transcriptional module discovery by jointly modeling gene expression and ChIP-chip data. *BMC Bioinformatics*, **8**, 283.

Liu, Y., Qiao, N., Zhu, S., Su, M., Sun, N., Boyd-Kirkup, J., and Han, J.-D. J. (2013). A novel Bayesian network inference algorithm for integrative analysis of heterogeneous deep sequencing data. *Cell Research*, **23**(3), 440–443.

Lkhagvasuren, O. (2010). *Integrierte Analyse von ChIP - Chip und ChIP - Seq Daten*. Bachelor's thesis, Faculty of Statistics, TU Dortmund University.

Lloyd, S. P. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, **28**, 129–137.

Lo, A. Y. (1984). On a class of Bayesian nonparametric estimates: I. density estimates. *The Annals of Statistics*, **12**(1), 351–357.

Lopez-Kleine, L., Leal, L., and Lopez, C. (2013). Biostatistical approaches for the reconstruction of gene co-expression networks based on transcriptomic data. *Briefings in Bioinformatics*, **12**(5), 457–467.

Louhimo, R., Lepikhova, T., Monni, O., and Hautaniemi, S. (2012). Comparative analysis of algorithms for integration of copy number and expression data. *Nature Methods*, **9(4)**(4), 351–355.

MacEachern, S. and Müller, P. (1998). Estimating mixture of Dirichlet process models. *Journal of Computational and Graphical Statististics*, **7**, 223–238.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, page 281–297. University of California Press.

Mahling, M., Höhling, M., and Küchenhoff, H. (2013). Determinig high-risk zones for unexploded World War II bombs by using point process methodology. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, **62**(2), 181–199.

Maitra, R. and Ramler, I. P. (2009). Clustering in the presence of scatter. *Biometrics*, **65**, 341–352.

Manley, S., Gillette, J. M., Patterson, G. H., Shroff, H., Hess, H. F., Betzig, E., and Lippincott-Schwartz, J. (2008). High-density mapping of single-molecule trajectories with photoactivated localization microscopy. *Nature Methods*, **5**(2), 155–157.

Marioni, J. C., Mason, C. E., Mane, S. M., Stephens, M., and Gilad, Y. (2008). RNA-seq: An assessment of technical reproducibility and comparison with gene expression arrays. *Genome Research*, **18**(9), 1509–1517.

MATLAB (2010). *Version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts.

Matérn, B. (1960). *Spatial variation*, volume 49. Meddelanden från Statens Skogsforskningsinstitut.

Matérn, B. (1986). *Spatial variation*. Berlin, Springer, 2nd edition.

McCarthy, D. J., Chen, Y., and Smyth, G. K. (2012). Differential expression analysis of multifactor RNA-seq experiments with respect to biological variation. *Nucleic Acids Research*, **40**, 4288–4297.

McCubrey, J. A., Steelman, L. S., Chappell, W. H., Abrams, S. L., Montalto, G., Cervello, M., Nicoletti, F., Fagone, P., Malaponte, G., Mazzarino, M. C., Candido, S., Libra, M., Bäsecke, J., Mijatovic, S., Maksimovic-Ivanic, D., Milella, M., Tafuri, A., Cocco, L., Evangelisti, C., Chiarini, F., and Martelli, A. M. (2012). Mutations and deregulation of Ras/Raf/MEK/ERK and PI3K/PTEN/Akt/mTOR cascades which alter therapy response. *Oncotarget*, **3**(9), 954–987.

McEachern, S. N. (1999). Dependent nonparametric processes. In *ASA Proceedings of the Section on Bayesian Statistical Science*, pages 50–55. American Statistical Association, Alexandria.

McLachlan, G. and Krishnan, T. (2008). *The EM algorithm and extensions*. Wiley, second edition.

McLachlan, G. and Peel, D. (2000). *Finite mixture models.* Wiley, New York.

McLeay, R., Lesluyes, T., Cuellar-Partida, G., and Bailey, T. (2012). Genome-wide in silico prediction of gene expression. *Bioinformatics*, **28**, 2789–2796.

Medvedovic, M. and Sivaganesan, S. (2002). Bayesian infinite mixture model based clustering of gene expression profiles. *Bioinformatics*, **18**(9), 1194–1206.

Medvedovic, M., Yeung, K., and Bumgarner, R. (2004). Bayesian mixture model based clustering of replicated microarray data. *Bioinformatics*, **20**, 1222–1232.

Mehlen, P., Delloye-Bourgeois, C., and Chédotal, A. (2011). Novel roles for Slits and netrins: axon guidance cues as anticancer targets? *Nature Reviews Cancer*, **11**(3), 188–197.

Mendoza-Parra, M.-A., Nowicka, M., van Gool, W., and Gronemeyer, H. (2013). Characterising ChIP-seq binding patterns by model-based peak shape deconvolution. *BMC Genomics*, **14**(1), 834.

Mitra, R. and Müller, P. (2014). Hierarchical Bayesian models for ChIP-seq data. In S. Datta and D. Nettleton, editors, *Statistical analysis of next generation sequencing data*, pages 297–314. Springer.

Mo, Q. (2012). A fully Bayesian hidden Ising model for ChIP-seq data analysis. *Biostatistics*, **13**(1), 113–128.

Mockler, T. C. and Ecker, J. R. (2005). Applications of DNA tiling arrays for whole-genome analysis. *Genomics*, **85**, 1–15.

Mortazavi, A., Williams, B. A., McCue, K., Schaeffer, L., and Wold, B. (2008). Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nature Methods*, **5**, 621–628.

Moyal, J. E. (1962). The general theory of stochastic population processes. *Acta Matematica*, **108**, 1–31.

Muliere, P. and Secchi, P. (1995). A note on a proper Bayesian bootstrap. Technical Report 18, Università degli Sudi di Pavia, Dipartimento di Economia Politica e Metodi Quantitativi.

Müller, P., Quintana, F., and Rosner, G. (2004). A method for modeling combining inference across related nonparametric Bayesian models. *Journal of the Royal Statistical Society B*, **66**, 735–749.

Müller, P., Quintana, F., and Rosner, G. (2011). A product partition model with regression on covariates. *Journal of Computational and Graphical Statistics*, **20**(1), 260–278.

Müller-Heine, A. (2009). *Simulation und Detektion von Ras-Protein-Clustern auf der Zellmembran*. Diploma thesis, Faculty of Statistics, TU Dortmund University.

Nair, N. U., Sahu, A. D., Bucher, P., and Moret, B. M. E. (2012). ChIPnorm: a statistical method for normalizing and identifying differential regions in histone modification ChIP-seq libraries. *PLoS One*, **7**(8), e39573.

Nan, X., Collisson, E., Lewis, S., Huang, J., Tamgüney, T. M., Liphardt, J. T. McCormick, F., Gray, J. W., and Chu, S. (2013). Single-molecule superresolution imaging allows quantitative analysis of RAF multimer formation and signaling. *Proceedings of the National Academy of Sciences of the United States of America*, **110**(46), 18519–18524.

Neal, R. M. (2000). Markov chain sampling methods. *Journal of Computational and Graphical Statistics*, **9**(2), 249–265.

Needham, C., Bradford, J., Bulpitt, A., and Westhead, D. (2006). Inference in bayesian networks. *Nature Biotechnology*, **24**, 51–53.

Newton, M. A., Noueiry, A., Sarkar, D., and Ahlquist, P. (2004). Detecting differential gene expression with a semiparametric hierarchical mixture method. *Biostatistics*, **5**(2), 155–176.

Neyman, J. and Scott, E. L. (1958). A statistical approach to problems of cosmology. *Journal of the Royal Statistical Society B*, **20**, 1–43.

Ongaro, A. and Cattaneo, C. (2004). Discrete random probability measures: A general framework for nonparametric Bayesian inference. *Statistics and Probability Letters*, **67**, 33–45.

Ortiz-Estevez, M., De las Rivas, J., Fontanillo, C., and Rubio, A. (2011). Segmentation of genomic and transcriptomic microarrays data reveals major correlation between DNA copy number aberrations and gene–loci expression. *Genomics*, **97**, 86–93.

Oshlack, A. and Wakefield, M. J. (2009). Transcript length bias in RNA-seq data confounds systems biology. *Biology Direct*, **4**, 14.

Ouyang, Z., Zhou, Q., and Wong, W. (2009). ChIP-Seq of transcription factors predicts absolute and differential gene expression in embryonic stem cells. *Proceedings of the National Academy of Sciences of the United States of America*, **106**, 21521–21526.

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

Park, S. and Nakai, K. (2011). A regression analysis of gene expression in es cells reveals two gene classes that are significantly different in epigenetic patterns. *BMC Bioinformatics*, **12(Suppl.1)**, S50.

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

Pepke, S., Wold, B., and Mortazavi, A. (2009). Computation for ChIP-seq and RNA-seq studies. *Nature Methods*, **6**, S22 – S32.

Pique-Regi, R., Degner, J. F., Pai, A. A., Gaffney, D. J., Gilad, Y., and Pritchard, J. K. (2011). Accurate inference of transcription factor binding from DNA sequence and chromatin accessibility data. *Genome Research*, **21**, 447–455.

Pitman, J. (1996). Some developments of the Blackwell-MacQueen urn scheme. In T. S. Ferguson, L. S. Shapeley, and J. MacQueen, editors, *Statistics, Probability and Game Theory. Papers in Honor of David Blackwell*, pages 245–268. Haywar, California: IMS Lecture Notes - Monograph Series.

Pitman, J. and Yor, M. (1997). The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *The Annals of Probability*, **2**, 855–900.

Plowman, S. J., Muncke, C., Parton, R. G., and Hancock, J. (2005). H-ras, K-ras, and inner plasma membrane raft proteins operate in nanoclusters with differential dependence on the actin cytoskeleton. *Proceedings of the National Academy of Sciences of the United States of America*, **102**(43), 15500–15505.

Prior, I. A. and Hancock, J. F. (2012). Ras trafficking, localization and compartmentalized signalling. *Seminars in Cell and Developmental Biology*, **23**, 145–153.

Qin, J., Hu, Y., Xu, F., Yalamanchili, H., and Wang, J. (2014). Inferring gene regulatory networks by integrating ChIP-seq/chip and transcriptome data via LASSO-type regularization methods. *Methods*, **67**, 294–303.

Qin, Z. S., Yu, J., Shen, J., Maher, C. A., Hu, M., Kalyana-Sundaram, S., Yu, J., and Chinnaiyan, A. M. (2010). HPeak: an HMM-based algorithm for defining read-enriched regions in ChIP-Seq data. *BMC Bioinformatics*, **11**, 369.

Quintana, F. (2006). A predictive view of Bayesian clustering. *Journal of Statistical Planning and Inference*, **136**, 2407–2429.

Quintana, F. A. and Iglesias, P. L. (2003). Bayesian clustering and product partition models. *Journal of the Royal Statistical Society B*, **65**, 557–574.

R Core Team (2011). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

R Core Team (2014). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Rapaport, F., Khanin, R., Liang, Y., Pirun, M., Krek, A., Zumbo, P., Mason, C. E., Socci, N. D., and Betel, D. (2013). Comprehensive evaluation of differential gene expression analysis methods for RNA-seq data. *Genome Biology*, **14**, R95.

Rashid, N. U., Giresi, P. G., Ibrahim, J. G., Sun, W., and Lieb, J. D. (2011). ZINBA integrates local covariates with DNA-seq data to identify broad and narrow regions of enrichment, even within amplified genomic regions. *Genome Biology*, **12**, R67.

Rasmussen, C. E., De la Cruz, B. J., Ghahramani, Z., and Wild, D. L. (2009). Modeling and visualizing uncertainty in gene expression clusters using Dirichlet process mixtures. *IEEE Transactions on Computational Biology and Bioinformatics*, **6**(4), 615–628.

Rau, A., Celeux, G., Martin-Magniette, M.-L., and Maugis-Rabusseau, C. (2011). Clustering high-throughput sequencing data with Poisson mixture models. Research Report RR-7786.

Reiss, D. J., Facciotti, M. T., and Baliga, N. S. (2007). Model-based deconvolution of genome-wide DNA binding. *Bioinformatics*, **24**(3), 396–403.

Ren, L., Du, L., Carin, L., and Dunson, D. B. (2011). Logistic stick-breaking process. *Journal of Machine Learning Research*, **12**, 203–239.

Richardson, S. and Green, P. J. (1997). On Bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society B*, **59**(4), 731–792.

Ripley, B. D. (1977). Modelling spatial patterns. *Journal of the Royal Statistical Society B*, **39**(2), 172–212.

Ripley, B. D. and Kelly, F. P. (1977). Markov point processes. *Journal of the London Mathematical Society*, **15**, 188–192.

Ritchie, M., Holzinger, E., Li, R., Pendergrass, S., and Kim, D. (2015). Methods of integrating data to uncover genotype-phenotype interactions. *Nature Reviews Genetics*, **16**(2), 85–97.

Robertson, G., Hirst, M., Bainbridge, M., Bilenky, M., Zhao, Y., Zeng, T., Euskirchen, G., Bernier, B., Varhol, R., Delaney, A., Thiessen, N., Griffith, O., He, A., Marra, M., Snyder, M., and Jones, S. (2007). Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. *Nature Methods*, **4**, 651–657.

Robinson, M. D. and Smyth, G. K. (2007). Moderated statistical tests for assessing differences in tag abundance. *Bioinformatics*, **23**, 2881–2887.

Robinson, M. D., McCarthy, D. J., and Smyth, G. K. (2010). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, **26**(1), 139–140.

Rodriguez, A. (2008). The nested Dirichlet process (with discussion). *Journal of the American Statistical Association*, **103**(483), 1131–1154.

Rodriguez, A. and Dunson, D. B. (2011). Nonparametric Bayesian models through probit stick-breaking processes. *Bayesian Analysis*, **6**(1), 145–178.

Rodriguez, A., Dunson, D. B., and Gelfand, A. E. (2010). Latent stick-breaking processes. *Journal of the American Statistical Association*, **105**(490), 647–658.

Roeder, K. and Wasserman, L. (1997). Practical Bayesian density estimation using mixtures of normals. *Journal of the American Statistical Association*, **92**, 894–902.

Rogers, A. (1974). *Statistical analysis of spatial dispersion*. Pion, London.

Rotblat, B., Belanis, L., Liang, H., Haklai, R., Elad-Zefadia, G., F. Hancock, J., Kloog, Y., and Plowman, S. (2010). H-ras nanocluster stability regulates the magnitude of MAPK signal output. *PloS one*, **5**, e11991.

Rousseau, J. and Mengersen, K. (2011). Asymptotic behaviour of the posterior distribution in overfitted mixture models. *Journal of the Royal Statistical Society B*, **73**(4), 689–710.

Rozowsky, J., Euskirchenn, G., Auerbach, R. K., Zhang, Z. D., Gibson, T., Bjornson, R., Carriero, N., Snyder, M., and Gerstein, M. B. (2009). PeakSeq: systematic scoring of ChIP-Seq experiments relative to controls. *Nature Biotechnology*, **27**(1), 66–75.

Saffarian, S., Li, Y., Elson, E., and Pike, L. (2007). Oligomerization of the EGF receptor investigated by live cell fluorescence intensity distribution analysis. *Biophysical Journal*, **93**(3), 1021–1031.

Savage, R. S., Ghahramani, Z., Griffin, J. E., De la Cruz, B. J., and Wild, D. L. (2010). Discovering transcriptional modules by Bayesian data integration. *Bioinformatics*, **26**(12), i158–i167.

Schäfer, M. and Ickstadt, K. (2012). An algorithm to quantify the clustering behaviour of Ras proteins in the plasma membrane. Forschungsbericht 2012/1, TU Dortmund University.

Schäfer, M., Schwender, H., Merk, S., Haferlach, C., Ickstadt, K., and Dugas, M. (2009). Integrated analysis of copy number alterations and gene expression: a bivariate assessment of equally directed abnormalities. *Bioinformatics*, **25**(24), 3228–3235.

Schäfer, M., Lkhagvasuren, O., Klein, H.-U., Elling, C., Wüstefeld, T., Müller-Tidow, C., Zender, L., Koschmieder, S., Dugas, M., and Ickstadt, K. (2012). Integrative analyses for Omics data: A Bayesian mixture model to assess the concordance of ChIP-chip and ChIP-seq measurements. *Journal of Toxicology and Environmental Health, Part A*, **75**, 461–470.

Schäfer, M., Radon, Y., Klein, T., Herrmann, S., Schwender, H., Verveer, P., and Ickstadt, K. (2015). A Bayesian mixture model to quantify parameters of spatial clustering. *Computational Statistics and Data Analysis*, **92**, 163–176.

Scharl, T. and Leisch, F. (2006). The stochastic QT-clust algorithm: Evaluation of stability and variance on time-course microarray data. In A. Rizzi and M. Vichi, editors, *Compstat 2006-Proceedings in Computational Statistics*, pages 1015–1022. Physica Verlag, Heidelberg, Germany.

Schenk, T., Chen, W. C., Göllner, S., Howell, L., Jin, L., Hebestreit, K., Klein, H.-U., Popescu, A. C., Burnett, A., Mills, K., Casero, R. A., Marton, L., Woster, P., Minden, M. D., Dugas, M., Wang, J. C. Y., Dick, J. E., Müller-Tidow, C., Petrie, K., and Zelent, A. (2012). Inhibition of the LSD1 (KDM1A) demethylase reactivates the all-trans-retinoic acid differentiation pathway in acute myeloid leukemia. *Nature Medicine*, **18**(4), 605–611.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, **6**, 461–464.

Schweikert, G., Cseke, B., Clouaire, T., Bird, A., and Sanguinetti, G. (2013). MMDiff: quantitative testing for shape changes in ChIP-Seq data sets. *BMC Genomics*, **14**, 826.

Schwender, H. (2007). *Statistical analysis of genotype and gene expression data*. Ph.D. thesis, Department of Statistics of the University of Dortmund.

Selinski, S. and Ickstadt, K. (2008). Cluster analysis of genetic and epidemiological data in molecular epidemiology. *Journal of Toxicology and Environmental Health A*, **71**(11-12), 835–844.

Sengupta, P., Jovanovic-Talisman, T., Skoko, D., Renz, M., Veatch, S., and Lippincott-Schwartz, J. (2011). Probing protein heterogeneity in the plasma membrane using PALM and pair correlation analysis. *Nature Methods*, **8**(11), 969–975.

Sengupta, P., Jovanovic-Talisman, T., and Lippincott-Schwartz, J. (2013). Quantifying spatial organization in point-localization superresolution images using pair correlation analysis. *Nature Protocols*, **8**(2), 345–354.

Seyednasrollah, F., L., A., and Elo, L. L. (2013). Comparison of software packages for detecting differential expression in RNA-seq studies. *Briefings in Bioinformatics*.

Seyednasrollah, F., Laiho, A., and Elo, L. L. (2015). Comparison of software packages for detecting differential expression in RNA-seq studies. *Briefings in Bioinformatics*, **16**(1), 59–70.

Sha, K. and Boyer, L. A. (2009). The chromatin signature of pluripotent cells. stemBook, URL: http://www.ncbi.nlm.nih.gov/books/NBK27041/. last accessed on 02.07.2015.

Shalom-Feuerstein, R., Plowman, S. J., Rotblat, B., Ariotti, N., Tian, T., Hancock, J. F., and Kloog, Y. (2008). K-ras nanoclustering is subverted by overexpression of the scaffold protein galectin-3. *Cancer Research*, **68**(16), 6608–6616.

Shamimuzzaman, M. and Vodkin, L. (2013). Genome-wide identification of binding sites for NAC and YABBY transcription factors and co-regulated genes during soybean seedling development by ChIP-Seq and RNA-Seq. *BMC Genomics*, **14**, 477.

Shi, Y., Chinnaiyan, A., and Jiang, H. (2015). rSeqNP: A non-parametric approach for detecting differential expression and splicing from RNA-Seq data. *Bioinformatics*, **31**(13), 2222–2224.

Shroff, H., Galbraith, C. G., Galbraith, J., White, H., Gillette, J., Olenych, S., Davidson, M. W., and Betzig, E. (2007). Dual-color superresolution imaging of genetically expressed probes within individual adhesion complexes. *Proceedings of the National Academy of Sciences of the United States of America*, **104**(51), 20308–20313.

Si, Y. and Liu, P. (2013). An optimal test with maximum average power while controlling FDR with application to RNA-seq data. *Biometrics*, **69**, 594–605.

Smyth, G. K. (2003). Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. *Statistical applications in genetics and molecular biology*, **3**(3), 1–25.

Smyth, G. K. (2005). Limma: linear models for microarray data. In R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, and W. Huber, editors, *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, pages 397–420. Springer, New York.

Soneson, C. and Delorenzi, M. (2013). A comparison of methods for differential expression analysis of RNA-seq data. *BMC Bioinformatics*, **14**, 91.

Song, Q. and Smith, A. D. (2011). Identifying dispersed epigenomic domains from ChIP-seq data. *Bioinformatics*, **27**(6), 870–871.

Spiegelhalter, D. J., Best, N. G., Carlin, B. P., and van der Linde, A. (2002). Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society, Series B*, **64**(4), 583–616.

Spiegelhalter, D. J., Thomas, A., Best, N. G., and Lunn, D. (2003). *WinBUGS User Manual, Version 1.4*.

Spyrou, C., Stark, R., Lynch, A. G., and Tavare, S. (2009). BayesPeak: Bayesian analysis of ChIP-seq data. *BMC Bioinformatics*, **10**, 299.

Srivastava, M. K., Khan, A. H., and Srivastava, N. (2014). *Statistical Inference: Theory of Estimation*. Prentice Hall India.

Srivastava, S. and Chen, L. (2010). A two-parameter generalized Poisson model to improve the analysis of RNA-seq data. *Nucleic Acids Research*, **38**(17), e170.

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

Stoyan, D. (1992). Statistical estimation of model parameters of planar Neyman-Scott cluster processes. *Metrika*, **39**(2), 67–74.

Stoyan, D. and Stoyan, H. (1994). *Fractals, random shapes and point fields: methods of geometrical Statistics*. Wiley, Chichester.

Strachan, T. and Read, A. (1999). *Human Molecular Genetics*. Wiley, second edition.

Strauss, D. J. (1975). A model for clustering. *Biometrika*, **63**, 467–475.

Svensén, M. and Bishop, C. M. (2004). Robust Bayesian mixture modelling. *Neurocomputing*, **64**, 235–252.

Tang, B., Hsu, H.-K., Hsu, P.-Y., Bonneville, R., Chen, S.-S., Huang, T. H.-M., and Jina, V. X. (2012). Hierarchical modularity in ER alpha transcriptional network is associated with distinct functions and implicates clinical outcomes. *Scientific Reports*, **2**, 875.

Tarazona, S., García-Alcalde, F., Dopazo, J., Ferrer, A., and Conesa, A. (2011). Differential expression in RNA-seq: a matter of depth. *Genome Res*, **21**, 2213–2223.

Taslim, C., Wu, J., Yan, P., Singer, G., Parvin, J., Huang, T., Lin, S., and Huang, K. (2009). Comparative study on ChIP-seq data: normalization and binding pattern characterization. *Bioinformatics*, **25**, 2334–2340.

Taslim, C., Lin, S., Huang, K., and Huang, T. H.-M. (2012). Integrative genome-wide chromatin signature analysis using finite mixture models. *BMC Genomics*, **13**, S3.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, **101**(476), 1566–1581.

Teicher, H. (1963). Identifiability of finite mixtures. *Annals of Mathematical Statistics*, **34**, 1265–1269.

Thomas, M. (1949). A generalization of Poisson's binomial limit for use in ecology. *Biometrika*, **36**, 18–25.

Tian, T., Harding, A., Inder, K., Plowman, S., Parton, R. G., and Hancock, J. (2007). Plasma membrane nanoswitches generate high-fidelity Ras signal transduction. *Nature Cell Biology*, **9**(8), 905–914.

Toomre, D. and Manstein, D. J. (2001). Lighting up the cell surface with evanescent wave microscopy. *Trends in Cell Biology*, **11**(7), 298–303.

Torra, V. (2005). Fuzzy c-means for fuzzy hierarchical clustering. In *Fuzzy Systems, 2005. FUZZ '05. The 14th IEEE International Conference on on Fuzzy Systems*, pages 646–651.

Trapnell, C., Williams, B. A., Pertea, G., Mortazavi, A., Kwan, G., van Baren, M. J., Salzberg, S. L., Wold, B. J., and Pachter, L. (2010). Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nature Biotechnology*, **28**(5), 511–515.

Trapnell, C., Hendrickson, D. G., Sauvageau, M., Goff, L., Rinn, J. L., and Pachter, L. (2013). Differential analysis of gene regulation at transcript resolution with RNA-seq. *Nature Biotechnology*, **31**(1), 46–53.

Tzikas, D. G., Likas, A., and Galatsanos, N. P. (2008). The variational approximation for Bayesian inference. *IEEE Signal Processing Magazine*, **145**, 131–146.

van de Wiel, M. A., Leday, G. G. R., Pardo, L., Rue, H., van der Vaart, A. W., and van Wieringen, W. N. (2012). Bayesian analysis of RNA sequencing data by estimating multiple shrinkage priors. *Biostatistics*, **14**, 113–128.

van de Wiel, M. A., de Menezes, R. X., Siebring-van Olst, E., and van Beusechem, V. W. (2013). Analysis of small-sample clinical genomics studies using multi-parameter shrinkage: application to high-throughput RNA interference screening. *BMC Medical Genomics*, **6**, 1.

van Lieshout, M. N. M. and Baddeley, A. J. (2002). Extrapolating and interpolating spatial patterns. In M. van Lieshout and A. Baddeley, editors, *Spatial cluster modeling*, pages 61–86. Chapman and Hall.

van Wieringen, W. N. and van de Wiel, M. A. (2009). Nonparametric testing for DNA copy number induced differential mRNA gene expression. *Biometrics*, **65**(1), 19–29.

Vega, V. B., Cheung, E., Palanisamy, N., and Sung, W.-K. (2009). Inherent signals in sequencing-based chromatin immunoprecipitation control libraries. *PLoS One*, **4**, e5241.

Wang, J., Lunyak, V., and Jordan, I. (2013). Broadpeak: a novel algorithm for identifying broad peaks in diffuse ChIP-seq datasets. *Bioinformatics*, **29**(4), 492–493.

Wang, L. and Dunson, D. B. (2011). Fast Bayesian inference in Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, **20**(1), 196–216.

Wang, L. and Wang, X. (2013). Hierarchical Dirichlet process model for gene expression clustering. *EURASIP Journal on Bioinformatics and Systems Biology*, **1**, 5.

Wang, N., Wang, Y., Hao, H., Wang, L., Wang, Z., Wang, J., and Wu, R. (2014). A bi-Poisson model for clustering gene expression profiles by RNA-seq. *Briefings in Bioinformatics*, **15**(4), 534–541.

Wang, P., Domeniconi, C., and Laskey, K. B. (2010). Nonparametric Bayesian clustering ensembles. In B. Balcázar, F. Bonchi, A. Gionis, and M. Sebag, editors, *Machine learning and knowledge discovery in databases*. Springer.

Wang, Y. R. and Huang, H. (2014). Review on statistical methods for gene network reconstruction using expression data. *Journal of Theoretical Biology*, **362**, 53–61.

Wang, Z., Gerstein, M., and Snyder, M. (2009). RNA-seq: a revolutionary tool for transcriptomics. *Nature Reviews Genetics*, **10**(1), 57–63.

Watson, D. M., Roshier, D. A., and Wiegand, T. (2007). Spatial ecology of a parasitic shrub: patterns and predictions. *Austral Ecology*, **32**, 359–369.

Wei, P. and Pan, W. (2008). Incorporating gene networks into statistical tests for genomic data via a spatially correlated mixture model. *Bioinformatics*, **24**, 404–411.

Wei, Y., Li, X., Wang, Q., and Ji, H. (2012). iASeq: integrative analysis of allele-specificity of protein-DNA interactions in multiple ChIP-seq datasets. *BMC Genomics*, **13**, 681.

Wellbrock, C., Karasarides, M., and Marais, R. (2004). The Raf proteins take centre stage. *Nature Reviews Molecular Cell Biology*, **5**(11), 875–885.

Wiegand, T., Gunatilleke, S., Gunatilleke, N., and Okuda, T. (2007). Analyzing the spatial structure of a Sri Lankan tree species with multiple scales of clustering. *Ecology*, **88**, 3088–3102.

Wiegand, T., Martínez, I., and Huth, A. (2009). Recruitment in tropical tree species: revealing complex spatial patterns. *The American Naturalist*, **174**(4), E106–E140.

Wikipedia (2014a). URL: http://de.wikipedia.org/wiki/Chromosomenmutation. last accessed on 02.07.2015.

Wikipedia (2014b). URL: http://en.wikipedia.org/wiki/Fluorescence_microscope. last accessed on 02.07.2015.

Wishart, D. (1969). Mode analysis: a generalization of nearest neighbor which reduces chaining effects. In A. Cole, editor, *Numerical Taxonomy*. Academic Press, London and New York.

Wolter, S., Schüttelpilz, M., Tscherepanow, M., van de Linde, S., Heilemann, M., and Sauer, M. (2010). Real-time computation of subdiffraction-resolution fluorescence images. *Journal of Microscopy*, **237**, 12 – 22.

Wolter, S., Löschberger, A., Holm, T., Aufmolk, S., Debauvalle, M.-C., van de Linde, S., and Sauer, M. (2012). rapidSTORM: accurate, fast open-source software for localization microscopy. *Nature Methods*, **9**(11), 1040–1041.

Wong, T.-T. (1998). General Dirichlet distribution in Bayesian analysis. *Applied Mathematics and Computation 97*, **97**, 165–181.

Wood, F., Goldwater, S., and Black, M. J. (2006). A non-parametric Bayesian approach to spike sorting. In *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 1, pages 1165–1168.

Wu, G. and Ji, H. (2013). ChIPXpress: using publicly available gene expression data to improve ChIP-seq and ChIP-chip target gene ranking. *BMC Bioinformatics*, **14**, 188.

Wu, H. and Ji, H. (2010). JAMIE: joint analysis of multiple ChIP-chip experiments. *Bioinformatics*, **26**(15), 1864–1870.

Xie, L., Weichel, B., Ohm, J., and Zhang, K. (2011). An integrative analysis of DNA methylation and RNA-Seq data for human heart, kidney and liver. *BMC Systems Biology*, **5(Suppl 3)**, S4.

Xing, H., Mo, Y., Liao, W., and Zhang, M. Q. (2012). Genome-wide localization of protein-DNA binding and histone modification by a Bayesian change-point method with ChIP-seq data. *PLoS Computational Biology*, **8**(7), e1002613.

Xu, H., Wei, C.-L., Lin, F., and Sung, W.-K. (2008). An HMM approach to genome-wide identification of differential histone modification sites from ChIP-seq data. *Bioinformatics*, **24**(20), 2344–2349.

Xu, J. and Zhang, Y. (2012). A generalized linear model for peak calling in ChIP-seq data. *Journal of Computational Biology*, **19**(6), 826–838.

Xu, M., Weinberg, C. R., Umbach, D. M., and Li, L. (2011). coMOTIF: a mixture framework for identifying transcription factor and a coregulator motif in ChIP-seq data. *Bioinformatics*, **27**(19), 2625–2632.

Xu, X., Hoang, S., Mayo, M., and Bekiranov, S. (2010). Application of machine learning methods to histone methylation ChIP-Seq data reveals H4R3me2 globally represses gene expression. *BMC Bioinformatics*, **11**, 396.

Xu, Y., Lee, J., Yuan, Y., Mitra, R., Liang, S., Müller, P., and Ji, Y. (2013). Nonparametric Bayesian bi-clustering for next generation sequencing count data. *Bayesian Analysis*, **8**(4), 759–780.

Yau, C. and Holmes, C. (2011). Hierarchical Bayesian nonparametric mixture models for clustering with variable relevance determination. *Bayesian Analysis*, **6**(2), 329–352.

Ye, M., Wang, Z., Wang, Y., and Wu, R. (2015). A multi-poisson dynamic mixture model to cluster developmental patterns of gene expression by RNA-seq. *Briefings in Bioinformatics*, **16**(2), 205–215.

Young, M. D., Wakefield, M. J., Smyth, G. K., and Oshlack, A. (2010). Gene ontology analysis for RNA-seq: accounting for selection bias. *Genome Biol.*, **11**, R14.

Yu, D., Huber, W., and Vitek, O. (2013). Shrinkage estimation of dispersion in negative binomial models for RNA-seq experiments with small sample size. *Bioinformatics*, **29**(10), 1275–1282.

Yu, H., Zhu, S., Zhou, B., Xue, H., and Han, J. (2008). Inferring causal relationships among different histone modifications and gene expression. *Genome Research*, **18**, 1314–1324.

Yu, K., Gong, B., Lee, M., Liu, Z., Xu, J., Perkins, R., and Tong, W. (2014). Discovering functional modules by topic modeling RNA-Seq based toxicogenomic data. *Chemical Research in Toxicology*, **27**, 1528–1536.

Zacher, B., Kuan, P. F., , and Tresch, A. (2010). Starr: Simple Tiling ARRay analysis of affymetrix ChIP-chip data. *BMC Bioinformatics*, **11**, 194.

Zang, C., Schones, D. E., Zeng, C., Cui, K., Zhao, K., and Peng, W. (2009). A clustering approach for identification of enriched domains from histone modification ChIP-Seq data. *Bioinformatics*, **25**(15), 1952–1958.

Zeng, X., Sanalkumar, R., Bresnick, E. H., Li, H., Chang, Q., and Keles, S. (2013). jMOSAiCS: joint analysis of multiple ChIP-seq datasets. *Genome Biology*, **14**, R38.

Zhang, L. and Mallick, B. K. (2013). Inferring gene networks from discrete expression data. *Biostatistics*, **14**(4), 708–722.

Zhang, L., Meng, J., Liu, H., and Huang, Y. (2012). A nonparametric Bayesian approach for clustering bisulfate-based DNA methylation profiles. *BMC Genomics*, **13(Suppl 6)**, S20.

Zhang, X., Bernatavichute, Y. V., Cokus, S., Pellegrini, M., and Jacobsen, S. E. (2009). Genome-wide analysis of mono-, di- and trimethylation of histone H3 lysine 4 in Arabidopsis thaliana. *Genome Biology*, **10**(6), R62.

Zhang, X., Robertson, G., Krzywinski, M., Ning, K., Droit, A., Jones, S., and Gottardo, R. (2011). PICS: probabilistic inference for ChIP-seq. *Biometrics*, **67**(1), 151–163.

Zhang, Y., Liu, T., Meyer, C. A., Eeckhoute, J., Johnson, D. S., Bernstein, B. E., Nusbaum, C., Myers, R. M., Brown, M., Li, W., and Liu, X. S. (2008). Model-based analysis of ChIP-seq (MACS). *Genome Biology*, **9**(9), R137.

Zhao, K., Lu, Z.-X., Park, J. W., Zhou, Q., and Xing, Y. (2013). GLiMMPS: robust statistical model for regulatory variation of alternative splicing using RNA-seq data. *Genome Biology*, **14**, R74.

Zhou, X., Lindsay, H., and Robinson, M. D. (2014). Robustly detecting differential expression in RNA sequencing data using observation weights. *Nucleic Acids Research*, **42**(11), e91.

Zhou, Y.-H., Xia, K., and Wright, F. A. (2011). A powerful and flexible approach to the analysis of RNA sequence count data. *Bioinformatics*, **27**(19), 2672–2678.

Zhu, L. J. (2013). Integrative analysis of ChIP-chip and ChIP-seq dataset. *Methods in Molecular Biology*, **1067**, 105–124.