Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Master's Thesis in Computer Science

Model order selection: Criteria, inference strategies and an application to biclustering

Patrick Pletscher*

Machine Learning Group at the
Computer Science Department, ETH Zürich

March 5th, 2007 – September 4th, 2007

Supervised by:
Peter Orbanz, Prof. Joachim M. Buhmann

*Email: pat@pletscher.org

Preface

This thesis is submitted to the Department of Computer Science in partial fulfillment of the requirements for the degree of Master of Science in Computer Science at ETH Zürich (Swiss Federal Institute of Technology Zurich).

In this thesis we study unsupervised clustering methods that select the number of clusters on their own. Traditional methods based on information theory, compare different models by penalizing more complicated models. More recently a sophisticated method, known as the Dirichlet process has been applied to clustering problems; one of its biggest advantages is the theoretical sound foundation: we have one model for different number of clusters. This however comes at a price, too: The inference is arguably even harder than for "standard" clustering models, but in recent years researchers proposed approximation algorithms that run efficiently, but sacrifice accuracy to a certain extent. In this thesis we aim to empirically compare these algorithms on synthetic data. We also compare the results with algorithms stemming from different motivations than the Dirichlet process, such as the Akaike information criterion (AIC) or the Bayesian information criterion (BIC).

In the second part we then study the application of the Dirichlet process to the problem of biclustering and propose two novel nonparametric algorithms, each of them assuming a different problem formulation. The two algorithms might also prove to be useful for feature selection and dimensionality reduction.

Keywords: Dirichlet process, probabilistic inference, Gibbs sampling, variational inference, model order selection, finite mixture model, Bayesian information criterion (BIC), Akaike information criterion (AIC), biclustering, nonparametric Bayesian modelling, feature selection.

Acknowledgments

First and above all, I want to express my gratitude to Peter Orbanz; during the course of this master's thesis he always took the time to answer my questions and he gave me valuable input how to improve certain experiments or express facts more concisely. He has the great gift of explaining things in an easy-to-understand way, which however doesn't sacrifice correctness. I could already witness that in the machine learning courses I attended as part of my studies where Peter was a teaching assistant. Then I also would like to thank Prof. Buhmann, for being my mentor and leveraging my interest in machine learning during my master's studies. He was very supportive in finding a topic

that suits my interests and knowledge, and also left a certain degree of freedom, to see where the journey takes us. Also, his comments in various meetings were very helpful in refining the biclustering models.

I was fortunate enough to work half a year as an intern under the supervision of Matthew Brand at the Mitsubishi Electric Research Labs (MERL) in Cambridge, MA. Matt, possessing an immense knowledge of such diverse areas such as machine learning, graphics, computer vision or theoretical computer science, could usually answer questions I was thinking about for half a day, within seconds, which taught me a lot about abstracting things away. Most importantly the time with MERL gave me a deep insight into research in general, but beside that I learned a lot of things that were directly or indirectly applicable to my master's thesis. I want to thank Matt for making this possible. I enjoyed the time in and around Boston, also because of all the great interns and MERL employees that I met there.

I want to thank Volker Roth for his short, but immensely valuable comment at my midterm presentation of the thesis, where he was suggesting to concentrate on the likelihood as a MOS quality measure, instead of expected number of clusters and such like.

I want to thank Charles Kemp for sending me the animal-feature matrix of a psychological experiment [Osherson et al., 1991], the data set is used for the evaluation of our biclustering algorithm.

This work is dedicated to my family and friends. I'd like to thank my parents and my brother for the enormous support during my studies at ETH. I am also grateful to my friends for reminding me to not forget about other, non-academic (and probably more important) things in life, such as travelling or sports (even though the Wifu team hardly ever won a soccer game).

Patrick Pletscher
Zürich, Switzerland
September, 2007

Mathematical Notation and Symbols

I have tried to keep this thesis report self-contained by reviewing important concepts of probability theory and related fields in chapter 2, this should allow an interested reader with a basic knowledge of linear algebra and calculus, as typically taught as part of an undergraduate degree in engineering or science, to understand a fair part of this documentation. However, for some of the more advanced topics discussed, a certain exposure to concepts of pattern recognition, machine learning, probability & statistics and algorithms is definitely recommended.

Also, I have tried to use a consistent notation throughout the documentation, this might however deviate from the notation in some of the publications cited or standard literature. Column vectors are always denoted by lowercase bold letters such as $\boldsymbol{x}$. A superscript T denotes the transpose of a matrix or a vector. Uppercase bold letters, such as $\boldsymbol{M}$, denote matrices. To denote the j-th component of a vector $\boldsymbol{x}$, we use x_j.

If we have N values $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ of a d-dimensional vector, we will usually combine the observations into a data matrix $\boldsymbol{X}$ in which the n-th row corresponds to the row vector $\boldsymbol{x}_n^T$. Also, we define the scalar product as $\langle \cdot, \cdot \rangle := \langle \cdot, \cdot \rangle_{\mathbb{R}^d} + \langle \cdot, \cdot \rangle_{\mathbb{R}^{d \times d}}$, where the scalar product over vectors is the traditional inner product and the scalar product over matrices is defined to be the trace of the matrix, i.e. $\langle \boldsymbol{A}, \boldsymbol{B} \rangle_{\mathbb{R}^{d \times d}} = \operatorname{tr}(\boldsymbol{A} \cdot \boldsymbol{B})$; for symmetric matrices this is the same as the scalar product of the vectorized matrices.

Probability and Statistics

Symbol	Meaning
Ω	sample space
$\mathcal{A}, \mathcal{B}$	σ-fields, mostly Borel σ-fields
$P(\cdot)$	probability mass function
$p(\cdot)$	probability density function; e.g. the Gaussian
$\mathcal{N}(\boldsymbol{x}\|\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$\mathrm{Dir}(\boldsymbol{x}\|\boldsymbol{\alpha})$	Dirichlet distribution with parameter $\boldsymbol{\alpha}$.
$\mathrm{Mult}(\boldsymbol{x}\|\boldsymbol{\mu})$	Multinomial distribution with bin probabilities $\boldsymbol{\mu}$
Z	normalization constant for densities
$\mathrm{E}_p[X]$	expected value of a random variable X, w.r.t. distribution p
$\sim$	distributed according to; e.g. $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
$D(q\|\|p)$	Kullback-Leibler divergence of distributions q and p
G_0	prior distribution
F	likelihood distribution
G	distribution sampled from a Dirichlet process

Clustering and model order selection

Symbol	Meaning
$\boldsymbol{\theta}_n$	cluster component of data point $\boldsymbol{x}_n$
$\boldsymbol{\theta}_k^*$	cluster component of cluster k, where $k = 1, \ldots, N_C$
N_C	number of clusters
N	number of samples
d	dimensionality of the data
$\boldsymbol{X}$	data, dimension $N \times d$
z_n	assignment of sample n to a cluster
$\pi_1, \ldots, \pi_{N_C}$	mixing proportions of a mixture model
m_k	number of samples assigned to cluster k
$\mathrm{DP}(\alpha, G_0)$	Dirichlet process with concentration α and base measure G_0
$\kappa(N_C)$	number of free parameters of a model with N_C factors

The notation $\kappa(N_C)$ is missleading, as the complexity of a model of course does not only depend on the number of clusters, but also on the model itself, i.e. for a Gaussian where we estimate a full covariance matrix and the mean, one cluster has complexity $d + d^2$. We dropped the conditioning on the model for not cluttering the notation.

Biclustering

Symbol	Meaning
$N_C^{\mathcal{O}}$	number of object clusters
$\mathcal{O}_1, \ldots, \mathcal{O}_{N_C^{\mathcal{O}}}$	clustering of the objects
$N_C^{\mathcal{F}}$	number of feature clusters
$\mathcal{F}_1, \ldots, \mathcal{F}_{N_C^{\mathcal{F}}}$	clustering of the features
$m_\mu^{\mathcal{O}}$	number of samples assigned to object cluster μ
$m_\nu^{\mathcal{F}}$	number of samples assigned to feature cluster ν

Contents

1 Introduction

> Sometimes it's hard to know where I stand. It's hard to know where I am. Well maybe it's a puzzle I don't understand.
>
> *(Keane in 'Is It Any Wonder')*

One of the most basic tasks in machine learning aims to find a classifier $\boldsymbol{f}$, that given an input $\boldsymbol{X}$ in $\mathbb{R}^{N\times d}$ outputs a labeling $\boldsymbol{y}$ in $\mathbb{N}^N$. There exist many different approaches to build such a classifier; they can broadly be divided into *supervised* (labeling of training set is available and used in order to learn $\boldsymbol{f}$) and *unsupervised* (where this labeling is not available/used). Supervised algorithms include traditional methods such as neural networks or linear discriminant analysis (LDA), but also more sophisticated approaches such as support vector machines (SVMs). Popular examples of unsupervised algorithms include k-means or the Gaussian mixture model (GMM) which is an example of an Expectation-Maximization (EM) algorithm; the different labels/classes are usually called clusters in unsupervised algorithms.

Traditional unsupervised algorithms however assume the number of clusters, e.g. the k in the k-means algorithm, is known. If k is not known, one uses *model order selection* (MOS) methods, such as the Bayesian information criterion (BIC). These approaches use a clustering algorithm for differing number of clusters and compare the likelihood, which should steadily increase when increasing k, and punish more complex models. To rephrase: these methods are looking for a trade-off between prediction performance (measured by the likelihood) and the complexity of the model, which increases the more clusters we add. While often used, they however have at least one conceptual flaw: we're comparing different models, which from a theoretical standpoint is not desirable: It would be best to have the model order selection included in the clustering model itself, and this is exactly what the Dirichlet process is all about! However little is known how the Dirichlet process performs compared to traditional MOS strategies. We try to address this in the first part of this master's thesis by a *comparative study of different MOS methods*.

In the second part of this master's thesis we will develop two novel nonparametric Bayesian *biclustering algorithms*, which are also evaluated on some synthetic data sets. In biclustering one has very similar problems as in standard clustering, like for example determining the number of clusters. However, as in biclustering, we are not only interested in a single clustering, but in two, the MOS question becomes arguably even more significant. We address this problem with a nonparametric approach, that shows

promising results.

In this thesis, we assume that the general big-picture approach is given by clustering. However, it should be pointed out, that there exist also different approaches for similar problems, which do not focus on clustering; a particularly interesting alternative is latent feature identification, which is especially worthwhile to mention, as it is also possible to define a nonparametric model by means of the Indian Buffet process [Griffiths and Ghahramani, 2005]. Both, the Dirichlet process and the Indian Buffet process try to address the MOS problem in a very similar way: they define a prior that assigns a non-zero probability to all possible clusterings (latent features), the probabilities are however, such that we can select a prior belief about the model order. We can then combine this prior, in a general Bayesian manner, with the observations to infer a solution.

One of the main tools, which we will extensively use throughout this thesis is given by *graphical models* and the inference algorithms that lie at the heart of such models. This is a subfield of machine learning that developed at a fast pace and is now ubiquitous. Graphical models are crucial, if the classification and learning problems are regarded as a coupled system. An intuitive example for such a coupling is weather prediction: if we had good weather during the past two weeks and today in the morning we see some smaller clouds in the sky, we would still assume it's gone be a sunny day; this might be different if in the past we often had unstable weather conditions; the bottom line being that the prediction is influenced by previous events which makes some random variables coupled.

1.1 Thesis Overview

Even Isac Newton remarked in a letter:

> If I have seen further it is by standing on the shoulders of giants.

However, to stand on the shoulders of giants we first have to climb them. For researchers this means crawling through the literature and understanding the fundamentals of the topic. We devote chapter 2 and chapter 3 to this: we review important concepts from machine learning in chapter 2 and discuss some of the inference algorithms for the Dirichlet process in chapter 3.

In chapter 4 we then compare the different methods for model order selection. This includes a theoretical study and an empirical discussion.

Finally, in chapter 5 we study an application of the Dirichlet process: We introduce two novel algorithms for biclustering.

1.2 Contributions

To the best of our knowledge nobody has ever seriously compared the Dirichlet process to its information-theoretic competitors used in the parametric setting, and thus chapter 4

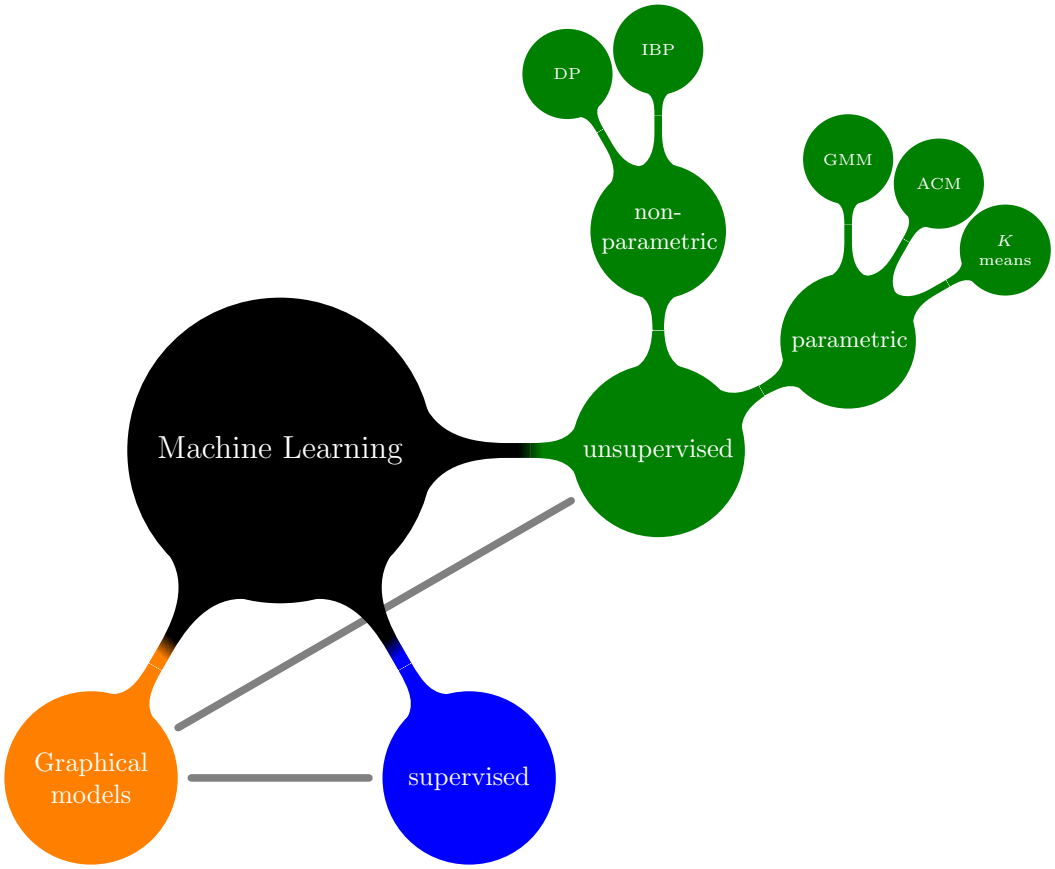


Figure 1.1: Schematic overview of some of the topics covered in this thesis.

is novel. Chapter 4 is mainly a collection of results from an experimental comparison between different MOS methods on synthetic data.

While not entirely novel, because MIT researchers proposed a similar model in the cognitive science community [Kemp et al., 2006], the biclustering algorithm introduced in chapter 5 should still be considered as a contribution to the field, as we consider arbitrary count data instead of binary data, which makes our approach more general. Also, we give an asymmetric biclustering algorithm, which could potentially be used for unsupervised feature selection.

2 Background

> Karma Police, arrest this man, he talks in maths.
> He buzzes like a fridge, he's like a detuned radio.
>
> *(Radiohead in 'Karma Police')*

In this chapter we introduce the mathematics that will be needed to understand the Dirichlet process, this includes probability theory and statistics (sections 2.1, 2.2 and 2.3) as well as graphical models (section 2.5). Furthermore we introduce more traditional clustering approaches, such as the EM algorithm in section 2.4. Researchers in the field might want to skip these sections and readily start with section 2.6 which introduces the Dirichlet process and go back to the preparative sections as needed.

2.1 Basics of Probability Theory

In this section we give a really short introduction to the theory of probability and statistics with a special focus towards afterwards introducing the Dirichlet process; this introduction neither claims to be complete nor to be fully self-contained (some basics from algebra and calculus are assumed to be known). In a first subsection we introduce some basic definitions from measure theory, thereafter we're ready to introduce probability, as it's a special instance of a measure. The definitions are taken from [Schervish, 1995].

2.1.1 Measure and Integration Theory

A *measure* is a way of assigning numerical values to the "sizes" of sets.

Definition 2.1. A nonempty subset $\mathcal{A}$ of the power set of a set Ω is called a *field* (or sometimes algebra) if

- $A \in \mathcal{A}$ implies $A^c \in \mathcal{A}$,
- $A_1, A_2 \in \mathcal{A}$ implies $A_1 \cup A_2 \in \mathcal{A}$.

A field $\mathcal{A}$ is called a σ-field if $\{A_i\}_{i=1}^{\infty} \in \mathcal{A}$ implies $\cup_{i=1}^{\infty} A_i \in \mathcal{A}$.

In other words if $\mathcal{A}$ is a σ-field, then it is nonempty, closed under complements and closed under countable unions.

Definition 2.2 (Borel σ-field)**.** Let $\mathcal{C}$ be the collection of intervals in $\mathbb{R}$. The smallest σ-field containing $\mathcal{C}$ is called the *Borel σ-field*.

Definition 2.3 (Measurable space)**.** A pair $(\Omega, \mathcal{A})$, where Ω is a set and $\mathcal{A}$ is a σ-field, is called a *measurable space*.

Definition 2.4 (Measure)**.** If $(\Omega, \mathcal{A})$ is a measurable space, then a function $\mu : \mathcal{A} \to [0, \infty]$ is called a *measure* if

- $\mu(\emptyset) = 0$,
- $\{A_i\}_{i=1}^{\infty}$ mutually disjoint implies $\mu(\cup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} \mu(A_i)$.

Definition 2.5 (Measure space)**.** If μ is a measure, the triple $(\Omega, \mathcal{A}, \mu)$ is called a *measure space*.

Definition 2.6 (Measurable function)**.** Suppose that Ω_1 is a set with a σ-field $\mathcal{A}_1$ of subsets, and let Ω_2 be another set with a σ-field $\mathcal{A}_2$ of subsets. Suppose that $f : \Omega_1 \to \Omega_2$ is a function. We say that f is *measurable* if for every $A \in \mathcal{A}_2$, $f^{-1}(A) \in \mathcal{A}_1$.

Theorem 2.7. *A measurable function f from one measure space $(\Omega_1, \mathcal{A}_1, \mu_1)$ to a measurable space $(\Omega_2, \mathcal{A}_2)$, $f : \Omega_1 \to \Omega_2$, induces a measure on the range Ω_2. For each $A \in \mathcal{A}_2$, define $\mu_2(A) = \mu_1(f^{-1}(A))$. Integrals with respect to μ_2 can be written as integrals with respect to μ_1 in the following way: If $g : \Omega_2 \to \mathbb{R}$ is integrable, then*

$$\int g(y) d\mu_2(y) = \int g(f(x)) d\mu_1(x).$$

Remark. The integration used here is an extension of the standard Riemann integral to integrate a function with respect to a measure.

Definition 2.8 (Induced Measure)**.** The measure μ_2 in Theorem 2.7 is called the *measure induced* on $(\Omega_2, \mathcal{A}_2)$ by f from μ_1.

2.1.2 Mathematical Probability

Definition 2.9. A *probability space* is a measure space $(\Omega, \mathcal{A}, \mu)$ with $\mu(\Omega) = 1$. Each element of $\mathcal{A}$ is called an *event*. If $(\Omega, \mathcal{A}, \mu)$ is a probability space, $(\mathcal{X}, \mathcal{B})$ is a measurable space, and $X : \Omega \to \mathcal{X}$ is measurable, then X is called a *random quantity*. If $\mathcal{X} = \mathbb{R}$ and $\mathcal{B}$ is the Borel σ-field, then X is called a *random variable*. Let μ_X be the probability measure induced on $(\mathcal{X}, \mathcal{B})$ by X from μ (see Definition 2.8). This probability measure is called *the distribution of X*. The distribution of X is said to be *discrete* if there exists a countable set $A \subseteq \mathcal{X}$ such that $\mu_X(A) = 1$. The distribution of $\mathcal{X}$ is *continuous* if $\mu_X(\{x\}) = 0$ for all $x \in \mathcal{X}$.

Example 2.10. Let $\Omega = \mathcal{X} = \mathbb{R}$ with Borel σ-field. Let p be a nonnegative function such that $\int p(x)\ dx = 1$. Define $\mu(A) = \int_A p(x) dx$ and $X(s) = s$. Then X is a continuous random variable with *probability density function* p, and $\mu_X = \mu$.

Example 2.11. Let $\Omega = \mathbb{R}$ with Borel σ-field. Let $\mathcal{X} = \{x_1, x_2, \ldots\}$, a countable set. Let P be a nonnegative function defined on $\mathcal{X}$ such that $\sum_{i=1}^{\infty} P(x_i) = 1$. Define $\mu(A) = \sum_{\{i : x_i \in A\}} P(x_i)$. Then X is a discrete random variable with *probability mass function* P, and $\mu_X = \mu$.

2.1.3 Exchangeability

Definition 2.12 (Exchangeable)**.** A finite set $X_1, \ldots, X_n$ of random quantities is said to be *exchangeable* if every permutation of $(X_1, \ldots, X_n)$ has the same joint distribution as every other permutation. An infinite collection is exchangeable if every finite subcollection is exchangeable.

Theorem 2.13 (DeFinetti's representation theorem)**.** *Let $(\Omega, \mathcal{A}, \mu)$ be a probability space, and let $(\mathcal{X}, \mathcal{B})$ be a Borel space. For each i, let $X_i : \Omega \to \mathcal{X}$ be measurable. The sequence $\{X_i\}_{i=1}^{\infty}$ is exchangeable if and only if there is a random probability measure $\boldsymbol{P}$ on $(\mathcal{X}, \mathcal{B})$ such that, conditional on $\boldsymbol{P} = p$, $\{X_i\}_{i=1}^{\infty}$ are i.i.d. with distribution p. Furthermore, if the sequence is exchangeable, then the distribution of $\boldsymbol{P}$ is unique, and $\boldsymbol{P}_i(B)$ converges to $\boldsymbol{P}(B)$ almost surely for each $B \in \mathcal{B}$.*

What the DeFinetti Theorem tells us is, that if $X_1, X_2, \ldots$ are infinitely exchangeable then the joint probability $p(x_1, x_2, \ldots)$ has a representation as a mixture:

$$p(x_1, x_2, \ldots) = \int_{\Omega_\theta} \left(\prod_{i=1}^{\infty} p(x_i|\theta) \right) dP(\theta),$$

for some random variable θ. This is illustrated in Figure 2.1.

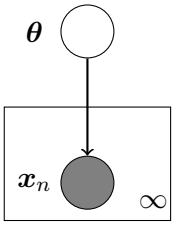


Figure 2.1: The DeFinetti Theorem as a graphical model.

While in this section we used capital letters to differentiate random variables from the concrete realization, we won't do this in the remainder of this thesis, mainly because we will require a lot of variables and parameters and it would be a cumbersome task to also differentiate between random variables and their realizations.

2.2 Important Probability Distributions

In this section we introduce some probability distributions, which we will use extensively in the remainder of this thesis, these are: the Gaussian, the Multinomial, the Dirichlet distribution and in the end a whole class of distributions, called the exponential family.

2.2.1 The Gaussian distribution

The Gaussian, also known as the normal distribution, is a widely used model for the distribution of continuous variables. Its distribution for a d-dimensional vector $\boldsymbol{x}$ is given

by

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) := \frac{1}{Z_{\mathcal{N}}(\boldsymbol{\Sigma})} \exp\left\{ -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \right\}, \tag{2.1}$$

where $\boldsymbol{\mu}$ is a d-dimensional mean vector, $\boldsymbol{\Sigma}$ is a $d \times d$ covariance matrix, and the normalization constant $Z_{\mathcal{N}}(\boldsymbol{\Sigma})$ is defined as follows

$$Z_{\mathcal{N}}(\boldsymbol{\Sigma}) := (2\pi)^{d/2} \det(\boldsymbol{\Sigma})^{1/2}.$$

The Gaussian distribution has several important properties, we however won't delve into these, as it is beyond the scope of this report. Below in Figure 2.2 we show samples from 2 different Gaussians for $d = 2$.

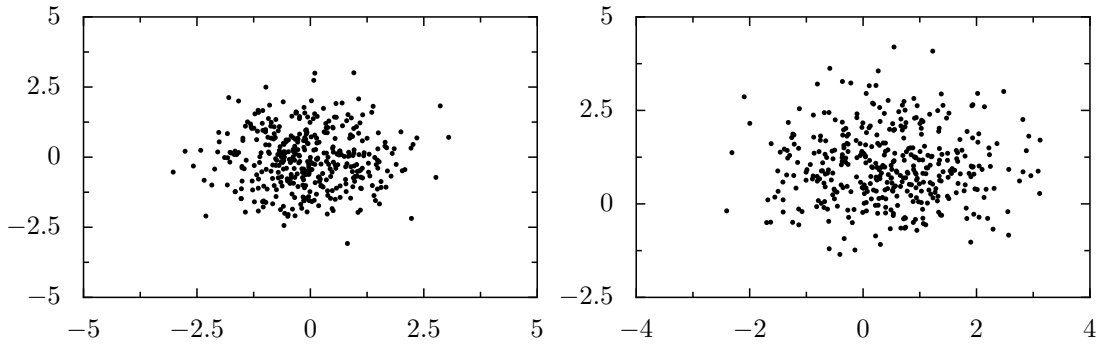


Figure 2.2: Samples from the Gaussian distribution for different means and covariances.

2.2.2 The Multinomial distribution

Given K bins where bin k occurs with probability μ_k. The probabilities μ_k have to follow the constraints $\mu_k \geq 0$ (non-negativity) and $\sum_{k=1}^{K} \mu_k = 1$ (normalization) and can be represented as a vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_K)^T$. Note that because of the summation constraint, the distribution of the $\{\mu_k\}$ is confined to a *simplex* of dimensionality $K-1$. Now let's imaging drawing M times one of the bins with the corresponding probability; this is a generalization of the Bernoulli variable, where we have $K = 2$. Again, we can represent the outcome as a vector $\boldsymbol{x}$, where x_k for $1 \leq k \leq K$ reflects the number of times bin k was drawn. The probability distribution of $\boldsymbol{x}$ is then given by a Multinomial distribution:

$$\text{Mult}(\boldsymbol{x}|\boldsymbol{\mu}, M) := \frac{M!}{x_1! \cdots x_K!} \prod_{k=1}^{K} \mu_k^{x_k} \tag{2.2}$$

This is for example a popular distribution in natural language processing (NLP) as there researchers often use the bag of words assumption: the order of the words within a document doesn't matter. A document can then be represented by a Multinomial variable where K is the size of the vocabulary and M the size of the document (with

the stop words removed).

Sometimes we are only interested to sample a single assignment instead of a vector, we then slightly abuse the notation by writing $\text{Mult}(z|\boldsymbol{\mu}, 1)$, then $z = k$ with probability μ_k.

2.2.3 The Dirichlet distribution

The Dirichlet distribution is a well-known prior distribution for the parameter $\boldsymbol{\mu}$ of the Multinomial distribution; it can be seen as a distribution over the parameters of a Multinomial distribution, as the resulting random variable $\boldsymbol{x}$ (of dimension K) is constrained to the $K-1$ simplex. The Beta distribution is a special case of the Dirichlet distribution for $K = 2$. The distribution is given by

$$\text{Dir}(\boldsymbol{x}|\boldsymbol{\alpha}) := \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)\cdots\Gamma(\alpha_K)} \prod_{k=1}^{K} x_k^{\alpha_k - 1}. \tag{2.3}$$

Here $\Gamma(x)$ is the Gamma function

$$\Gamma(x) := \int_0^\infty u^{x-1} e^{-u}\, du.$$

For an integer x the equality $\Gamma(x+1) = x!$ holds. Furthermore we denote the sum of the elements of $\boldsymbol{\alpha}$ by α_0:

$$\alpha_0 = \sum_{k=1}^{K} \alpha_k.$$

While as mentioned, $\boldsymbol{x}$ lies in the simplex of dimensionality $K-1$, $\boldsymbol{\alpha}$ doesn't necessarily have to obey such constraints. However, it's still useful to think about it as a normalized vector, let's introduce

$$\boldsymbol{\alpha}' = \boldsymbol{\alpha}/\alpha_0.$$

Like this we can interpret α_0 as the number of pseudo-measurements observed to obtain $\boldsymbol{\alpha}'$. We see a concentration phenomena: the more observations we measure, the more our confidence in $\boldsymbol{\alpha}$ and hence the more peaked the Dirichlet distribution around $\boldsymbol{\alpha}'$. In a Bayesian interpretation, the Dirichlet mean $\boldsymbol{\alpha}'$ could be said to be associated with a prior belief α_0. This is illustrated below in Figure 2.3: the bigger the value of α_0, the smaller the scatter of $\boldsymbol{x}$ and vice versa.

2.2.4 The Exponential Family

All of the distributions discussed so far in this section (and many more) are all members of a general class of probability models called exponential families. It is useful to study this generic class, as the members share many important properties and deriving them in general makes the cumbersome and tedious individual derivation for each member redundant.

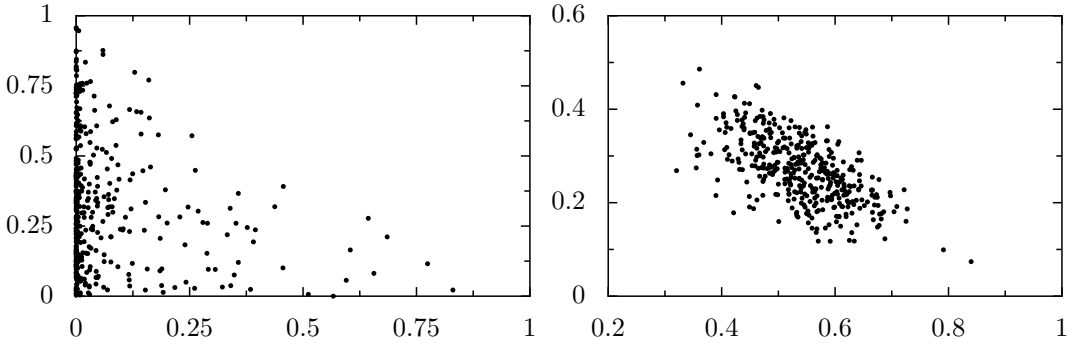


Figure 2.3: The Dirichlet distribution (we show the first and second dimension of $\boldsymbol{x}$) for different values of $\boldsymbol{\alpha}$. Left: $\boldsymbol{\alpha} = [0.2, 1, 2]$, right: $\boldsymbol{\alpha} = [20, 10, 7]$.

The exponential family of distributions over $\boldsymbol{x}$, given parameters $\boldsymbol{\lambda}$, is defined to be the set of distributions of the form

$$p(\boldsymbol{x}|\boldsymbol{\lambda}) = h(\boldsymbol{x}) \exp\{\langle \boldsymbol{\lambda}, \boldsymbol{s}(\boldsymbol{x}) \rangle - a(\boldsymbol{\lambda})\}, \tag{2.4}$$

where $\boldsymbol{x}$ may be scalar or vector, and may be discrete or continuous. Here $\boldsymbol{\lambda}$ are called the *natural parameters* of the distribution, and $\boldsymbol{s}(\boldsymbol{x})$ are the *sufficient statistics* of $\boldsymbol{x}$. The coefficient $a(\boldsymbol{\lambda})$ is a coefficient that ensures that the distribution is normalized and therefore satisfies

$$a(\boldsymbol{\lambda}) = \ln\left(\int_{\Omega_{\boldsymbol{x}}} h(\boldsymbol{x}) \exp\{\langle \boldsymbol{\lambda}, \boldsymbol{s}(\boldsymbol{x}) \rangle\} \, d\boldsymbol{x} \right),$$

which is usually called the *log partition function* (or sometimes cumulant generating function).

Using the general property of exponential families, that

$$\mathrm{E}[\boldsymbol{\lambda}] = \nabla_{\boldsymbol{\lambda}} a(\boldsymbol{\lambda}),$$

we can compute expectations without integrating.

Next, we show that the Gaussian, the Multinomial and Dirichlet distributions are all members of the exponential family.

Gaussian distribution

Let's study the Gaussian distribution, given by

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) := \frac{1}{Z_{\mathcal{N}}(\boldsymbol{\Sigma})} \exp\left\{ -\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{x} - \boldsymbol{\mu}) \right\}.$$

To bring this into the exponential form of (2.4), we introduce parameters as follows:

$$\begin{aligned}\boldsymbol{\lambda} &= [\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}, \boldsymbol{\Sigma}^{-1}] \\ \boldsymbol{s}(\boldsymbol{x}) &= [\boldsymbol{x}, -\frac{1}{2}\boldsymbol{x}\boldsymbol{x}^T] \\ a(\boldsymbol{\lambda}) &= \frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln(\det(\boldsymbol{\lambda}_2)) + \frac{1}{2}\boldsymbol{\lambda}_1^T\boldsymbol{\lambda}_2^{-1}\boldsymbol{\lambda}_1 \\ h(\boldsymbol{x}) &= 1.\end{aligned}$$

Where $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$ denote the two different parts of the natural parameter vector $\boldsymbol{\lambda}$, i.e. $\boldsymbol{\lambda}_1 = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$ and $\boldsymbol{\lambda}_2 = \boldsymbol{\Sigma}^{-1}$.

Multinomial distribution

Let's study the Multinomial distribution, given by

$$\text{Mult}(\boldsymbol{x}|\boldsymbol{\mu}, M) := \frac{M!}{x_1! \cdots x_K!} \prod_{k=1}^{K} \mu_k^{x_k}.$$

To bring this into the exponential form of (2.4), we introduce parameters as follows:

$$\begin{aligned}\boldsymbol{\lambda} &= [\ln(\mu_1), \ldots, \ln(\mu_K)]^T \\ \boldsymbol{s}(\boldsymbol{x}) &= [x_1, \ldots, x_K]^T \\ a(\boldsymbol{\lambda}) &= 0 \\ h(\boldsymbol{x}) &= \frac{(\sum_{k=1}^{K} x_k)!}{x_1! \cdots x_K!}.\end{aligned}$$

Dirichlet distribution

Let's study the Dirichlet distribution, given by

$$\text{Dir}(\boldsymbol{x}|\boldsymbol{\alpha}) := \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1) \cdots \Gamma(\alpha_K)} \prod_{k=1}^{K} x_k^{\alpha_k - 1}.$$

To bring this into the exponential form of (2.4), we introduce parameters as follows:

$$\begin{aligned}\boldsymbol{\lambda} &= [\alpha_1 - 1, \ldots, \alpha_K - 1]^T \\ \boldsymbol{s}(\boldsymbol{x}) &= [\ln(x_1), \ldots, \ln(x_K)]^T \\ a(\boldsymbol{\lambda}) &= \sum_{k=1}^{K} \ln(\Gamma(\lambda_k + 1)) - \ln\Gamma\left(\sum_{k=1}^{K} \lambda_k + K\right) \\ h(\boldsymbol{x}) &= 1.\end{aligned}$$

2.3 Conjugate Priors

In general, for a given likelihood distribution $F(\boldsymbol{x}|\boldsymbol{\theta})$, we can seek a prior $G_0(\boldsymbol{\theta})$ that is conjugate to the likelihood function, so that the posterior distribution has the same functional form as the prior. As a remainder, the posterior is given by the likelihood times the prior normalized by the evidence:

$$p(\boldsymbol{\theta}|\boldsymbol{x}) = \frac{F(\boldsymbol{x}|\boldsymbol{\theta})G_0(\boldsymbol{\theta})}{\int_{\Omega_{\boldsymbol{\theta}}} F(\boldsymbol{x}|\boldsymbol{\theta})G_0(\boldsymbol{\theta})\ d\boldsymbol{\theta}}.$$

In this section we derive conjugate priors for two particularly interesting distributions. First we show that the Gaussian with known mean and covariance is conjugate to a Gaussian likelihood with known covariance, second we prove that the Dirichlet distribution is conjugate to the Multinomial distribution.

The concept of conjugate priors is useful, as typical quantities like the evidence or the posterior needed for sampling algorithms, can efficiently be sampled from and computed, as long as efficient sampling algorithms for the prior are available; we'll come back to this in section 3.1.

To make it clear to the reader which distribution is used as a prior and which one as a likelihood we use a different symbol for the two: the likelihood is always denoted by F, while the prior is denoted by G_0. Often the likelihood and prior have hyperparameters which we haven't included in our notation for reasons of clarity; for example a Gaussian prior has hyperparameters $\boldsymbol{\mu}_\theta$ and $\boldsymbol{\Sigma}_\theta$.

2.3.1 Gaussian/Gaussian

Assuming both, the likelihood and the prior have a multivariate Gaussian distribution as introduced in subsection 2.2.1, i.e.

$$F(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{\Sigma}_x) = \frac{1}{Z_{\mathcal{N}}(\boldsymbol{\Sigma}_x)} \exp\left\{ -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\theta})^T \boldsymbol{\Sigma}_x^{-1}(\boldsymbol{x}-\boldsymbol{\theta}) \right\},$$

and

$$G_0(\boldsymbol{\theta}|\boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta) = \frac{1}{Z_{\mathcal{N}}(\boldsymbol{\Sigma}_\theta)} \exp\left\{ -\frac{1}{2}(\boldsymbol{\theta}-\boldsymbol{\mu}_\theta)^T \boldsymbol{\Sigma}_\theta^{-1}(\boldsymbol{\theta}-\boldsymbol{\mu}_\theta) \right\}.$$

The parameters $\boldsymbol{\Sigma}_x, \boldsymbol{\mu}_\theta$ and $\boldsymbol{\Sigma}_\theta$ are assumed to be given. In the remainder we show that the conjugate prior of the Gaussian distribution is a Gaussian distribution, too.

The posterior is given by

$$
\begin{aligned}
&p(\boldsymbol{\theta}|\boldsymbol{x}_{1:N}, \boldsymbol{\Sigma}_x, \boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta) \\
&\qquad \propto \left(\prod_{n=1}^{N} F(\boldsymbol{x}_n|\boldsymbol{\theta}, \boldsymbol{\Sigma}_x)\right) G_0(\boldsymbol{\theta}|\boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta) \\
&\qquad \propto \exp\left\{-\frac{1}{2}\left(\sum_{n=1}^{N}(\boldsymbol{x}_n - \boldsymbol{\theta})^T\boldsymbol{\Sigma}_x^{-1}(\boldsymbol{x}_n - \boldsymbol{\theta}) + (\boldsymbol{\theta} - \boldsymbol{\mu}_\theta)^T\boldsymbol{\Sigma}_\theta^{-1}(\boldsymbol{\theta} - \boldsymbol{\mu}_\theta)\right)\right\} \\
&\qquad = \exp\left\{-\frac{1}{2}\left(\sum_{n=1}^{N}\boldsymbol{x}_n^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x}_n - 2\sum_{n=1}^{N}\boldsymbol{x}_n^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{\theta} + N\boldsymbol{\theta}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{\theta}\right.\right. \\
&\qquad\qquad \left.\left. + \boldsymbol{\theta}^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\theta} - 2\boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\theta} + \boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta\right)\right\} \\
&\qquad \propto \exp\left\{-\frac{1}{2}\boldsymbol{\theta}^T(N\boldsymbol{\Sigma}_x^{-1} + \boldsymbol{\Sigma}_\theta^{-1})\boldsymbol{\theta} + \sum_{n=1}^{N}\boldsymbol{x}_n^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{\theta} + \boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\theta}\right\} \\
&\qquad = \exp\left\{-\frac{1}{2}\boldsymbol{\theta}^T(N\boldsymbol{\Sigma}_x^{-1} + \boldsymbol{\Sigma}_\theta^{-1})\boldsymbol{\theta} + \left(\sum_{n=1}^{N}\boldsymbol{x}_n^T\boldsymbol{\Sigma}_x^{-1} + \boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\right)\boldsymbol{\theta}\right\}.
\end{aligned}
$$

In the computations above we made use of the fact, that $\boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta$ and $\sum_{n=1}^{N}\boldsymbol{x}_n^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x}_n$ are constant and we can thus neglect them. By comparison with (2.1) we can rewrite the expression on the last line to reveal the Gaussian distribution of the posterior,

$$
p(\boldsymbol{\theta}|\boldsymbol{x}_{1:N}, \boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_x, \boldsymbol{\Sigma}_\theta) = \frac{1}{Z_{\mathcal{N}}(\bar{\boldsymbol{\Sigma}})}\exp\left\{-\frac{1}{2}(\boldsymbol{\theta} - \bar{\boldsymbol{\mu}})^T\bar{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{\theta} - \bar{\boldsymbol{\mu}})\right\}, \tag{2.5}
$$

where

$$
\begin{aligned}
\bar{\boldsymbol{\Sigma}} &:= (N\boldsymbol{\Sigma}_x^{-1} + \boldsymbol{\Sigma}_\theta^{-1})^{-1}, \\
\bar{\boldsymbol{\mu}} &:= \bar{\boldsymbol{\Sigma}}(N\boldsymbol{\Sigma}_x^{-1}\hat{\boldsymbol{x}} + \boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta), \\
\hat{\boldsymbol{x}} &:= \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x}_n.
\end{aligned}
$$

A second quantity that will be of interest in later chapters is the evidence, i.e.

$$
\int_{\Omega_{\boldsymbol{\theta}}} F(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{\Sigma}_x)G_0(\boldsymbol{\theta}|\boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta)\, d\boldsymbol{\theta}.
$$

In the general non-conjugate case, the integral shown above is often intractable to compute, however for conjugate priors it is analytically solvable. Below we derive the solution for the Gaussian/Gaussian case. One important thing to realize, is that Bayes' Theorem allows us to rewrite the integral as follows,

$$
\int_{\Omega_{\boldsymbol{\theta}}} F(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{\Sigma}_x)G_0(\boldsymbol{\theta}|\boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta)\, d\boldsymbol{\theta} = \frac{F(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{\Sigma}_x)G_0(\boldsymbol{\theta}|\boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta)}{p(\boldsymbol{\theta}|\boldsymbol{x}, \boldsymbol{\Sigma}_x, \boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta)}.
$$

Although one can't see it directly from the equation above, this expression is independent of $\boldsymbol{\theta}$.

$$\begin{aligned}
&\frac{F(\boldsymbol{x}|\boldsymbol{\theta},\boldsymbol{\Sigma}_x)G_0(\boldsymbol{\theta}|\boldsymbol{\mu}_\theta,\boldsymbol{\Sigma}_\theta)}{p(\boldsymbol{\theta}|\boldsymbol{x},\boldsymbol{\Sigma}_x,\boldsymbol{\mu}_\theta,\boldsymbol{\Sigma}_\theta)} \\
&\qquad = Z\exp\Bigg\{-\frac{1}{2}\Bigg((\boldsymbol{x}-\boldsymbol{\theta})^T\boldsymbol{\Sigma}_x^{-1}(\boldsymbol{x}-\boldsymbol{\theta}) \\
&\qquad\qquad + (\boldsymbol{\theta}-\boldsymbol{\mu}_\theta)^T\boldsymbol{\Sigma}_\theta^{-1}(\boldsymbol{\theta}-\boldsymbol{\mu}_\theta) - (\boldsymbol{\theta}-\bar{\boldsymbol{\mu}})^T\bar{\boldsymbol{\Sigma}}^{-1}(\boldsymbol{\theta}-\bar{\boldsymbol{\mu}})\Bigg)\Bigg\} \\
&\qquad = Z\exp\Bigg\{-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x} + \boldsymbol{\theta}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\theta}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{\theta} - \frac{1}{2}\boldsymbol{\theta}^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\theta} + \boldsymbol{\theta}^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta \\
&\qquad\qquad - \frac{1}{2}\boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta + \frac{1}{2}\boldsymbol{\theta}^T\bar{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\theta} - \boldsymbol{\theta}^T\bar{\boldsymbol{\Sigma}}^{-1}\bar{\boldsymbol{\mu}} + \frac{1}{2}\bar{\boldsymbol{\mu}}^T\bar{\boldsymbol{\Sigma}}^{-1}\bar{\boldsymbol{\mu}}\Bigg\} \\
&\qquad = Z\exp\Bigg\{-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x} + \boldsymbol{\theta}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x} + \boldsymbol{\theta}^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta \\
&\qquad\qquad - \frac{1}{2}\boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta - \boldsymbol{\theta}^T\bar{\boldsymbol{\Sigma}}^{-1}\bar{\boldsymbol{\mu}} + \frac{1}{2}\bar{\boldsymbol{\mu}}^T\bar{\boldsymbol{\Sigma}}^{-1}\bar{\boldsymbol{\mu}}\Bigg\} \\
&\qquad = Z\exp\Bigg\{-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta + \frac{1}{2}\bar{\boldsymbol{\mu}}^T\bar{\boldsymbol{\Sigma}}^{-1}\bar{\boldsymbol{\mu}} \\
&\qquad\qquad + \boldsymbol{\theta}^T\Big(\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x} + \boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta - \bar{\boldsymbol{\Sigma}}^{-1}\bar{\boldsymbol{\Sigma}}(\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x} + \boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta)\Big)\Bigg\} \\
&\qquad = Z\exp\Bigg\{-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x} - \frac{1}{2}\boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta + \frac{1}{2}\bar{\boldsymbol{\mu}}^T\bar{\boldsymbol{\Sigma}}^{-1}\bar{\boldsymbol{\mu}}\Bigg\}.
\end{aligned}$$

Where we introduced $Z := Z_{\mathcal{N}}(\bar{\boldsymbol{\Sigma}})/(Z_{\mathcal{N}}(\boldsymbol{\Sigma}_x)Z_{\mathcal{N}}(\boldsymbol{\Sigma}_\theta))$ to simplify the equations. The terms that got dropped from the second to the third equation are equal to zero:

$$-\frac{1}{2}\boldsymbol{\theta}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{\theta} - \frac{1}{2}\boldsymbol{\theta}^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\theta} + \frac{1}{2}\boldsymbol{\theta}^T\bar{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\theta} = \frac{1}{2}\Big(-\boldsymbol{\theta}^T(\boldsymbol{\Sigma}_x^{-1}+\boldsymbol{\Sigma}_\theta^{-1})\boldsymbol{\theta} + \boldsymbol{\theta}^T\bar{\boldsymbol{\Sigma}}^{-1}\boldsymbol{\theta}\Big) = 0.$$

2.3.2 Multinomial/Dirichlet

Here we assume a Dirichlet prior

$$G_0(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)\dots\Gamma(\alpha_K)}\prod_{k=1}^{K}\theta_k^{\alpha_k-1},$$

and a Multinomial likelihood (for K different bins)

$$F(\boldsymbol{x}_{1:N}|\boldsymbol{\theta}) = \underbrace{\frac{M_1!}{x_{11}!\cdots x_{1K}!}\cdots\frac{M_N!}{x_{N1}!\cdots x_{NK}!}}_{=:Z}\prod_{k=1}^{K}\theta_k^{\hat{x}_k}.$$

Where as before in the section about the Multinomial distribution, x_{nk} represents the number of times bin k occurs in the data sample $\boldsymbol{x}_n$, furthermore we introduced $\hat{x}_k = \sum_{n=1}^{N} x_{nk}$ and let $\hat{M}$ denote the total number of draws, i.e. $\hat{M} = \sum_{k=1}^{K}\hat{x}_k$. Then the posterior has again the form of a Dirichlet distribution:

$$\begin{aligned} p(\boldsymbol{\theta}|\boldsymbol{x}_{1:N},\boldsymbol{\alpha}) &\propto \left(\prod_{n=1}^{N}F(\boldsymbol{x}_n|\boldsymbol{\theta})\right)G_0(\boldsymbol{\theta}|\boldsymbol{\alpha}) \\ &= Z\frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)\cdots\Gamma(\alpha_K)}\prod_{k=1}^{K}\theta_k^{\alpha_k+\hat{x}_k-1} \end{aligned}$$

We can determine the correct normalization coefficient by comparison with (2.3) to get

$$p(\boldsymbol{\theta}|\boldsymbol{x}_{1:N},\boldsymbol{\alpha}) = \frac{\Gamma(\alpha_0+\hat{M})}{\Gamma(\alpha_1+\hat{x}_1)\cdots\Gamma(\alpha_K+\hat{x}_K)}\prod_{k=1}^{K}\theta_k^{\alpha_k+\hat{x}_k-1}$$

As before for the Gaussian, we are also interested in the evidence. We can use the Bayes' trick to compute it:

$$\begin{aligned} \frac{F(\boldsymbol{x}|\boldsymbol{\theta})G_0(\boldsymbol{\theta}|\boldsymbol{\alpha})}{p(\boldsymbol{\theta}|\boldsymbol{x},\boldsymbol{\alpha})} &= \frac{\frac{M!}{x_1!x_2!\cdots x_K!}\prod_{k=1}^{K}\theta_k^{x_k}\frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)\cdots\Gamma(\alpha_K)}\prod_{k=1}^{K}\theta_k^{\alpha_k-1}}{\frac{\Gamma(\alpha_0+M)}{\Gamma(\alpha_1+x_1)\cdots\Gamma(\alpha_K+x_K)}\prod_{k=1}^{K}\theta_k^{\alpha_k+x_k-1}} \\ &= \frac{\Gamma(M+1)\Gamma(\alpha_0)\prod_{k=1}^{K}\Gamma(\alpha_k+x_k)}{\Gamma(\alpha_0+M)\prod_{k=1}^{K}\Gamma(x_k+1)\prod_{k=1}^{K}\Gamma(\alpha_k)}. \end{aligned}$$

A direct computer implementation (with fixed precision arithmetic) of the formula above would however result in an overflow even for quite small problems. A good trick to work around this issue, is to compute the logarithm of the evidence: this transforms all of the products into sums and the division becomes a subtraction, moreover the log of the Gamma function is implemented in many software packages and is much less likely to overflow than the Gamma function. In the end we have to exponentiate the result again.

2.3.3 Exponential Family

Suppose that both the likelihood and the prior are members of the exponential family. What is required for them to be conjugate? Assuming we observe N data points $\boldsymbol{x}_{1:N}$, all being distributed according to the likelihood

$$F(\boldsymbol{x}_n|\boldsymbol{\theta}) = h(\boldsymbol{x}_n)\exp\Big\{\langle\boldsymbol{\theta},\boldsymbol{s}(\boldsymbol{x}_n)\rangle - a(\boldsymbol{\theta})\Big\},$$

then the prior has to have the following form:

$$G_0(\boldsymbol{\theta}|\boldsymbol{\lambda},\chi) = \frac{1}{Z(\boldsymbol{\lambda},\chi)}\exp\Big\{\chi\langle\boldsymbol{\lambda},\boldsymbol{\theta}\rangle - \chi a(\boldsymbol{\theta})\Big\}.$$

Note, that $a(\cdot)$ in the exponent of the prior is *not* used as the log partition function, but instead $Z(\boldsymbol{\lambda},\chi)$ is the partition function. Also, $\chi > 0$ has to hold. The posterior then has the form (where we've dropped the normalization constants $\prod_{n=1}^{N} h(\boldsymbol{x}_n)$ and $1/Z(\boldsymbol{\lambda},\chi)$),

$$p(\boldsymbol{\theta}|\boldsymbol{x}_{1:N},\boldsymbol{\lambda},\chi) \propto \exp\left\{\langle\boldsymbol{\theta},\sum_{n=1}^{N}\boldsymbol{s}(\boldsymbol{x}_n) + \chi\boldsymbol{\lambda}\rangle - a(\boldsymbol{\theta})(N+\chi)\right\}.$$

Like this it becomes evident, that χ can be seen as a prior belief, measuring our belief in the prior $\boldsymbol{\lambda}$. In this report we use the convention of multiplying the prior belief χ into the natural parameters, to get a posterior of the form

$$p(\boldsymbol{\theta}|\boldsymbol{x}_{1:N},\boldsymbol{\lambda},\chi) \propto \exp\left\{\langle\boldsymbol{\theta},\sum_{n=1}^{N}\boldsymbol{s}(\boldsymbol{x}_n) + \boldsymbol{\lambda}\rangle - a(\boldsymbol{\theta})(N+\chi)\right\},$$

where $\boldsymbol{\lambda}$ is now already appropriately scaled.

Gaussian/Gaussian

Expressing the Gaussian distribution as a member of the exponential family that is conjugate to another Gaussian distribution is slightly more involved than for the Dirichlet/Multinomial case; thus we list the computations here. Assuming a Gaussian prior (with mean $\boldsymbol{\mu}_\theta$ and covariance $\boldsymbol{\Sigma}_\theta$ and where we denote the parameter $\boldsymbol{\theta}$ from subsection 2.3.1 as $\boldsymbol{\mu}$, as we want to use $\boldsymbol{\theta}$ for the exponential family representation):

$$\begin{aligned} p(\boldsymbol{\mu}|\boldsymbol{\mu}_\theta,\boldsymbol{\Sigma}_\theta) &= \frac{1}{Z_{\mathcal{N}}(\boldsymbol{\Sigma}_\theta)}\exp\left\{-\frac{1}{2}\boldsymbol{\mu}^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu} + \boldsymbol{\mu}^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta - \frac{1}{2}\boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta\right\} \\ &= \underbrace{\frac{1}{Z_{\mathcal{N}}(\boldsymbol{\Sigma}_\theta)}\exp\left\{-\frac{1}{2}\boldsymbol{\mu}_\theta^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta\right\}}_{:=Z_1(\boldsymbol{\mu}_\theta,\boldsymbol{\Sigma}_\theta)}\exp\left\{\boldsymbol{\mu}^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta - \frac{1}{2}\boldsymbol{\mu}^T\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}\right\}, \end{aligned}$$

and a Gaussian likelihood (with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}_x$):

$$p(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}_x) = \underbrace{\frac{1}{Z_{\mathcal{N}}(\boldsymbol{\Sigma}_x)}\exp\left\{-\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x}\right\}}_{:=Z_2(\boldsymbol{x},\boldsymbol{\Sigma}_x)}\exp\left\{\boldsymbol{x}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{\mu} - \frac{1}{2}\boldsymbol{\mu}^T\boldsymbol{\Sigma}_x^{-1}\boldsymbol{\mu}\right\}.$$

We can represent this as an exponential family distribution by introducing

$$\boldsymbol{\lambda} = [\boldsymbol{\Sigma}_\theta^{-1}\boldsymbol{\mu}_\theta, \boldsymbol{\Sigma}_\theta^{-1}]$$

as the natural parameters of the prior and

$$\boldsymbol{\theta} = [\boldsymbol{\mu}, -\frac{1}{2}\boldsymbol{\mu}\boldsymbol{\mu}^T]$$

as the sufficient statistics. Then the prior can be written as

$$p(\boldsymbol{\theta}|\boldsymbol{\lambda}) = Z_1(\boldsymbol{\lambda}) \exp\{\langle \boldsymbol{\lambda}, \boldsymbol{\theta} \rangle\}, \tag{2.6}$$

with

$$Z_1(\boldsymbol{\lambda}) = \frac{\exp(-\frac{1}{2}\boldsymbol{\lambda}_1^T \boldsymbol{\lambda}_2^{-1} \boldsymbol{\lambda}_1)}{(2\pi)^{d/2} \det(\boldsymbol{\lambda}_2^{-1})^{1/2}}.$$

We can now introduce the sufficient statistics of the likelihood

$$\boldsymbol{x} = [\boldsymbol{\Sigma}_x^{-1}\boldsymbol{x}, \boldsymbol{\Sigma}_x^{-1}],$$

and the likelihood can then be written as

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = Z_2(\boldsymbol{x}) \exp\{\langle \boldsymbol{\theta}, \boldsymbol{x} \rangle\}$$

with

$$Z_2(\boldsymbol{x}) = \frac{\exp(-\frac{1}{2}\boldsymbol{x}_1^T \boldsymbol{x}_2^{-1} \boldsymbol{x}_1)}{(2\pi)^{d/2} \det(\boldsymbol{x}_{\boldsymbol{2}}^{-1})^{1/2}}.$$

2.4 Clustering and model order selection

In this section we discuss some of the classic clustering algorithms, such as Gaussian Mixture Model (GMM), Asymmetric Clustering Model (ACM) and the maximum-a-posteriori (MAP) versions thereof. These are all instances of a general class of algorithms called Expectation-Maximization methods. In the second part we then also discuss some traditional methods for inferring the number of clusters inherent in our data, this is usually referred to as model order selection (MOS).

2.4.1 Finite Mixture Model

A finite probability mixture model is characterized by a likelihood $F(\cdot)$, the number of distributions N_C, the mixing weights $\boldsymbol{\pi}$ (a probability vector of length N_C) and the components for each of the distributions $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}$. The probability density function is then a weighted combination of these different distributions:

$$p(\boldsymbol{x}|\boldsymbol{\pi}, \boldsymbol{\Theta}) = \sum_{k=1}^{N_C} \pi_k F(\boldsymbol{x}|\boldsymbol{\theta}_k^*).$$

This model can be expressed as a graphical model as shown in Figure 2.4.

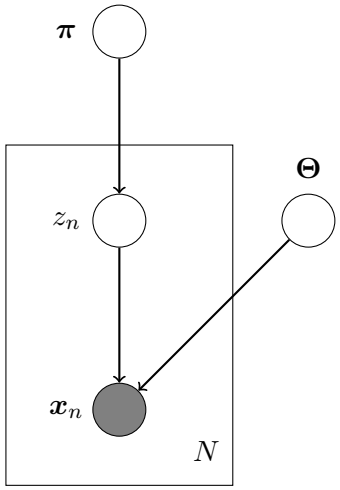


Figure 2.4: A finite mixture model as a graphical model; z_n denotes the cluster assignment of sample $\boldsymbol{x}_n$.

2.4.2 Expectation-Maximization

The expectation maximization algorithm, or EM algorithm, is a general technique for finding maximum likelihood solutions for probabilistic models having latent variables [Dempster et al., 1977]. For a general in-depth study, see for example [Bishop, 2007], section 9.4. We here mainly cover two special cases of the EM algorithm, which we will use later on in our experiments: the Gaussian mixture model and the asymmetric clustering model. The EM algorithm is also important because it is related to the variational inference framework discussed in section 2.5.

Gaussian Mixture Model (GMM)

For Gaussian data we assume that the hyperparameters of the prior ($\boldsymbol{\mu}_\theta$ and $\boldsymbol{\Sigma}_\theta$) and likelihood ($\boldsymbol{\Sigma}_x$) are known, which is of course an unlikely assumption for real world data, but as we are mainly interested in comparing different MOS models and clustering algorithms we feel that this a sensible choice. We thus only need to estimate the means of the clusters in the GMM clustering algorithm, in the general case one usually also estimates the covariance matrix for each cluster.

1. Initialize the means $\boldsymbol{\mu}_k$ and mixing coefficients π_k ($k = 1, \ldots, N_C$) and evaluate the initial value of the log likelihood.

2. **E step**. Evaluate the responsibilities using the current parameter values

$$q_{n,k} = \frac{\pi_k \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_x)}{\sum_{i=1}^{N_C} \pi_i \mathcal{N}(\boldsymbol{x}_n | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_x)}.$$

3. **M step**. Re-estimate the parameters using the current responsibilities

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} q_{n,k} \boldsymbol{x}_n$$
$$\pi_k = \frac{N_k}{N}$$

   where

$$N_k = \sum_{n=1}^{N} q_{n,k}.$$

4. Evaluate the log likelihood

$$\ln p(\boldsymbol{X}|\{\boldsymbol{\mu}\}_{k=1}^{N_C}, \boldsymbol{\Sigma}_x, \boldsymbol{\pi}) = \sum_{n=1}^{N} \ln \left\{ \sum_{k=1}^{N_C} \pi_k \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_x) \right\}$$

   and check for convergence of either the parameters or the log likelihood. If the convergence criterion is not satisfied return to step 2.

For incorporating the prior information we need to adapt the M step as follows:

$$\boldsymbol{\Sigma}_k = (N_k \boldsymbol{\Sigma}_x^{-1} + \boldsymbol{\Sigma}_\theta^{-1})^{-1}$$
$$\boldsymbol{\mu}_k = \boldsymbol{\Sigma}_k (N_k \boldsymbol{\Sigma}_x^{-1} \hat{\boldsymbol{x}}_k + \boldsymbol{\Sigma}_\theta^{-1} \boldsymbol{\mu}_\theta)$$

where

$$\hat{\boldsymbol{x}}_k := \frac{1}{N} \sum_{n=1}^{N} q_{n,k} \boldsymbol{x}_n.$$

Note that the common covariance gets replaced by a per cluster covariance matrix $\boldsymbol{\Sigma}_k$. For more explanations about the computation of the posterior, see subsection 2.3.1.

Asymmetric Clustering Model (ACM)

We use the ACM [Puzicha et al., 1999] algorithm for clustering histogram data, this algorithm is an instance of an EM algorithm and works as shown below. However in our experiments we used a MAP version of this algorithm with a Dirichlet prior to better compare it to other methods that assume a prior.

1. **E step**. Reestimate the responsibilities of histogram $\boldsymbol{x}_n$ belonging to cluster k:

$$q_{n,k} = \frac{\pi_k \text{Mult}(\boldsymbol{x}_n|\boldsymbol{\mu}_k)}{\sum_{i=1}^{N_C} \pi_i \text{Mult}(\boldsymbol{x}_n|\boldsymbol{\mu}_i)}$$

   where $\text{Mult}(\boldsymbol{x}|\boldsymbol{\mu})$ denotes the Multinomial distribution with factor $\boldsymbol{\mu}$.

2. **M step**. Reestimate the parameters:

$$\pi_k = \frac{\sum_{n=1}^{N} q_{n,k}}{\sum_{k=1}^{N_C}\sum_{n=1}^{N} q_{n,k}}$$

and

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} q_{n,k}\boldsymbol{x}_n}{\sum_{n=1}^{N} q_{n,k}}.$$

For a MAP version of this algorithm we have to add a prior term, given by the $\boldsymbol{\alpha}$ parameter of the Dirichlet distribution to the reestimation of the $\boldsymbol{\mu}_k$ parameters,

$$\boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} q_{n,k}\boldsymbol{x}_n + \boldsymbol{\alpha}}{\sum_{n=1}^{N} q_{n,k} + \alpha_0}.$$

2.4.3 Model order selection

Given two clusterings of the data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ with different number of clusters (for the same parametric form), which one is preferable? This is the question that lies at the heart of every model order selection strategy. While the Dirichlet process opts for a *local* criterion, in splitting the model for the n-th data point with a probability of $\alpha/(\alpha+n-1)$, traditional methods, such as the Akaike information criterion (AIC) [Akaike, 1974] and the Bayesian information criterion (BIC) [Schwarz, 1978], introduce a *global* criterion. The two scores for a given model order N_C are given by

$$\mathrm{AIC}_{N_C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N | \{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}, \boldsymbol{\pi}) := -\ln \ell_{N_C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N | \{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}, \boldsymbol{\pi}) + \kappa(N_C),$$

and

$$\mathrm{BIC}_{N_C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N | \{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}, \boldsymbol{\pi}) := -\ln \ell_{N_C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N | \{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}, \boldsymbol{\pi}) + \frac{1}{2}\kappa(N_C)\ln N.$$

Where $\{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}$ denotes all the latent factors, e.g. the Gaussian mean for every cluster. $\kappa(N_C)$ denotes the number of free parameters in our model and ℓ denotes the likelihood. We then want to find the model that leads to the minimal score. The BIC can be motivated as a Laplace approximation to the model evidence given by

$$p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N | N_C) = \int_{\Omega_{\boldsymbol{\Theta}}} p(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N | \boldsymbol{\Theta}, N_C) p(\boldsymbol{\Theta} | N_C)\ d\boldsymbol{\Theta}.$$

Where we collected all latent factors in $\boldsymbol{\Theta}$. For more information see e.g. section 4.4.1 in [Bishop, 2007]. In practice we assume we are given a lower and upper bound on the number of clusters and compute in a brute-force manner the score for every model order within the bound and in the end choose the clustering leading to the smallest score. We will come back to these information theoretic MOS strategies in section 4.1. It should be pointed out, that there exist other MOS approaches, such as stability [Lange et al., 2003] or minimum description length (MDL), which we will however not consider here.

2.5 Graphical Models and Probabilistic Inference

Graphical models are a standard tool to express dependencies and independencies of distributions. The topic is closely related to probabilistic inference, as normally what we are ultimately interested in, is finding a MAP estimate (or sometimes also marginals) for a given model and observations. In this section we give a very short, high-level overview of the concepts, for an in-depth discussion, see for example [Bishop, 2007, Jordan et al., 1999, Wainwright and Jordan, 2003].

2.5.1 Graphical Models

A graphical model is a graph, where each node has a random variable associated with it. The edges in this graph correspond to probabilistic dependencies of the random variables. Depending on the context, it is advantageous to either consider *directed* or *undirected* edges. In this thesis we will only consider directed graphical models (Bayesian networks), which is especially useful, if we think about the probabilities in a generative fashion, i.e. a variable can be expressed as a conditional probability, conditioned on other random variables. On the other hand, directed models (Markov random fields, conditional random fields) are important if we like to think about our problem in terms of energy minimization and/or factorization of the distribution into cliques.

2.5.2 Probabilistic inference in general

As exact inference for general graphical models is intractable, one usually considers approximations. These methods include belief propagation [Yedidia et al., 2000], variational inference [Jordan et al., 1999], sampling approaches [Neal, 1993], reparametrizations of the distribution [Wainwright et al., 2003] or graph-cut algorithms [Boykov et al., 2001]. In this thesis we only dealt with Gibbs sampling and variational inference and we thus restrict the discussion to these two approaches.

Gibbs sampling

Gibbs sampling is a rather simple Markov chain Monte Carlo algorithm. Assume we consider a distribution $p(\boldsymbol{z}) = p(z_1, \ldots, z_M)$ from which we wish to sample. Assume further, that we've chosen some initial configuration for the variables $z_1, \ldots, z_M$. Each step of the Gibbs sampler consists of drawing a new value for z_i based on the remaining values $\boldsymbol{z}_{-i}$ (which denotes $z_1, \ldots, z_M$ with z_i omitted). We repeat this step for $i = 1, \ldots, M$ or for some other (possibly random) order.

Variational inference

This part is taken from [Blei and Jordan, 2005]. Let's assume we are considering a model with hyperparameters $\boldsymbol{\vartheta}$, latent variables $\boldsymbol{W} = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M\}$, and observations $\boldsymbol{X}$. The posterior distribution of the latent variables is:

$$p(\boldsymbol{W}|\boldsymbol{X}, \boldsymbol{\vartheta}) = \exp\{\ln p(\boldsymbol{X}, \boldsymbol{W}|\boldsymbol{\vartheta}) - \ln p(\boldsymbol{X}|\boldsymbol{\vartheta})\}.$$

The log marginal probability of the data, needed to ensure the normalization of the distribution, is:

$$\ln p(\boldsymbol{X}|\boldsymbol{\vartheta}) = \ln \int p(\boldsymbol{W}, \boldsymbol{X}|\boldsymbol{\vartheta})\ d\boldsymbol{W},$$

which may be difficult to compute. Variational inference is based on reformulating the computation of the posterior distribution as an optimization problem, perturbing that problem and finding solutions to the perturbed problem. In this thesis we will only consider *mean-field* methods where we consider a family of distributions $q_{\boldsymbol{\nu}}(\boldsymbol{W})$, indexed by a variational parameter $\boldsymbol{\nu}$. We aim to minimize the KL-divergence between $q_{\boldsymbol{\nu}}$ and $p(\boldsymbol{W}|\boldsymbol{X}, \boldsymbol{\vartheta})$:

$$D(q_{\boldsymbol{\nu}}(\boldsymbol{W})\|p(\boldsymbol{W}|, \boldsymbol{X}, \boldsymbol{\vartheta})) = \mathrm{E}_q[\ln q_{\boldsymbol{\nu}}(\boldsymbol{W})] - \mathrm{E}_q[\ln p(\boldsymbol{W}, \boldsymbol{X}|\boldsymbol{\vartheta})] + \ln p(\boldsymbol{X}|\boldsymbol{\vartheta}),$$

where the problematic normalization constant does now not depend on the variational parameters and can thus be ignored for the optimization. We can alternatively also state the minimization problem above as a maximization of a lower bound on the log marginal likelihood:

$$\ln p(\boldsymbol{X}|\boldsymbol{\vartheta}) \geq \mathrm{E}_q[\ln p(\boldsymbol{W}, \boldsymbol{X}|\boldsymbol{\vartheta})] - \mathrm{E}_q[\ln q_{\boldsymbol{\nu}}(\boldsymbol{W})]. \tag{2.7}$$

For the optimization problem to be computationally tractable, we normally consider distributions $q_{\boldsymbol{\nu}}(\boldsymbol{W})$ where we broke some of the dependencies. We now consider such a family of distributions for exponential families. For each latent variable, let us assume that the conditional distribution $p(\boldsymbol{w}_i|\boldsymbol{W}_{-i}, \boldsymbol{X}, \boldsymbol{\vartheta})$ is a member of the exponential family:

$$p(\boldsymbol{w}_i|\boldsymbol{W}_{-i}, \boldsymbol{X}, \boldsymbol{\vartheta}) = h(\boldsymbol{w}_i) \exp\{\boldsymbol{g}_i(\boldsymbol{W}_{-i}, \boldsymbol{X}, \boldsymbol{\vartheta})^T \boldsymbol{w}_i - a(\boldsymbol{g}_i(\boldsymbol{W}_{-i}, \boldsymbol{X}, \boldsymbol{\vartheta}))\},$$

where $\boldsymbol{g}_i(\boldsymbol{W}_{-i}, \boldsymbol{X}, \boldsymbol{\vartheta})$ is the natural parameter for $\boldsymbol{w}_i$, when conditioning on the remaining variables and the observations.

For this setting Ghahramani and Beal [2001] propose to use the following family of distributions as mean-field variational approximations:

$$q_{\boldsymbol{\nu}}(\boldsymbol{W}) = \prod_{i=1}^{M} \exp\{\boldsymbol{\eta}_i^T \boldsymbol{w}_i - a(\boldsymbol{w}_i),$$

where $\boldsymbol{\nu} = \{\boldsymbol{\eta}_1, \ldots, \boldsymbol{\eta}_M\}$. The optimization of the KL divergence with respect to a single variational parameter ν_i, is achieved by computing the following expectation:

$$\boldsymbol{\eta}_i = \mathrm{E}_q[\boldsymbol{g}_i(\boldsymbol{W}_{-i}, \boldsymbol{X}, \boldsymbol{\vartheta})]. \tag{2.8}$$

2.6 The Dirichlet Process

In this section we introduce the *Dirichlet process* (DP) [Ferguson, 1973], which proposed back in the seventies, has become popular as a flexible clustering method only recently. We start by introducing the Dirichlet process as an abstract definition and afterwards show some of its properties, which arguably are more important and give more insights, than the definition itself.

Definition 2.14 (Dirichlet process)**.** Let $(\Omega, \mathcal{A})$ be a measurable space, with G_0 a probability measure on the space, and let α be a positive real number. The Dirichlet process is the distribution of a random probability measure G over $(\Omega, \mathcal{A})$ such that, for any finite partition $(A_1, \ldots, A_k)$ of Ω, the random vector $(G(A_1), \ldots, G(A_k))$ is distributed as a finite-dimensional Dirichlet distribution:

$$(G(A_1), \ldots, G(A_k)) \sim \mathrm{Dir}(\alpha G_0(A_1), \ldots, \alpha G_0(A_k)).$$

We write $G \sim \mathrm{DP}(\alpha, G_0)$ if G is a random probability measure distributed according to the Dirichlet process. We call G_0 the *base measure* of G and call α the *concentration parameter*.

We will often work with generative models in the DP setting, this usually boils down to: sample a distribution G from a Dirichlet process $\mathrm{DP}(\alpha, G_0)$ and afterwards sample N data points $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N$ from G. This process is usually denoted as follows:

$$\begin{aligned} G &\sim \mathrm{DP}(\alpha, G_0) \\ \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N &\sim G. \end{aligned} \tag{2.9}$$

Which is illustrated as a directed graphical model in Figure 2.5.

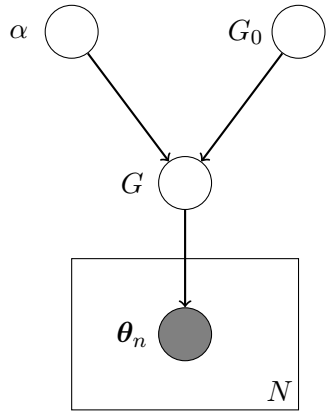


Figure 2.5: The Dirichlet process as a directed graphical model.

The Dirichlet process was specifically designed such that it allows for an efficient posterior update and as already proven in [Ferguson, 1973], the posterior process is again a Dirichlet process:

$$G \,|\, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N \sim \mathrm{DP}(\alpha, G_0 + \sum_{n=1}^{N} \delta_{\boldsymbol{\theta}_n}).$$

Where $\delta_{\boldsymbol{\theta}_n}$ is a Dirac measure centered at $\boldsymbol{\theta}_n$.

2.6.1 Stick-breaking construction and discreteness of the Dirichlet process

Measures G drawn from the Dirichlet Process as given in Definition 2.14 are discrete with probability one. This becomes most evident in the stick-breaking construction due to Sethuraman [1994]. Before formally introducing the stick-breaking construction we give an illustrating way to think about this representation which also led to its name. Let's assume we are given a stick of unit length and we break it into two pieces according to a Beta$(1, \alpha)$ distribution, let's denote the length of the resulting first part as π_1. We repeat this infinitely often with the remaining part and successively denote them as $\pi_2, \pi_3, \ldots$; this process is illustrated in Figure 2.6.

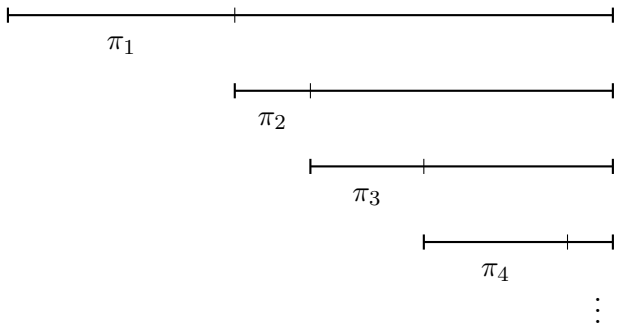


Figure 2.6: Breaking a stick infinitely often leads to the correct probability distribution for the cluster assignments.

These "stick lengths" are important for the construction given by Sethuraman [1994], which gives an explicit formulation of the Dirichlet process:

$$\begin{aligned} v_k &\sim \text{Beta}(1, \alpha) \\ \boldsymbol{\theta}_k^* &\sim G_0 \\ \pi_k(\boldsymbol{v}) &= v_k \prod_{i=1}^{k-1} (1 - v_i) \\ G &= \sum_{k=1}^{\infty} \pi_k(\boldsymbol{v}) \delta_{\boldsymbol{\theta}_k^*}. \end{aligned} \tag{2.10}$$

This representation makes clear that the distribution G from the DP is discrete with probability one; the support of G consists of a countably infinite set of atoms $\{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty}$, drawn independently from G_0. We here and later on use the convention of denoting distinct values by $\boldsymbol{\theta}^*$, the difference between $\boldsymbol{\theta}$ and $\boldsymbol{\theta}^*$ should become obvious in the next subsection.

2.6.2 Sampling from a Dirichlet process and the Pólya urn scheme

Instead of giving an explicit representation of G, as done in the previous subsection, one might be interested in the draws of the DP, as this would then allow us to sample from

G. The Pólya urn scheme [Blackwell and MacQueen, 1973] answers this question and shows also that the draws are discrete and exhibit a clustering effect.

Let $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots$ be a sequence of i.i.d. random variables distributed according to G. That is the variables $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots$ are conditionally independent given G, and hence exchangeable. Let us consider the successive conditional distributions of $\boldsymbol{\theta}_{n+1}$ given $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n$, where G has been integrated out. Blackwell and MacQueen [1973] showed that these conditional distributions have the following form:

$$\boldsymbol{\theta}_{n+1} \,|\, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n \sim \frac{n}{\alpha + n} \sum_{i=1}^{n} \delta_{\boldsymbol{\theta}_i}(\boldsymbol{\theta}_{n+1}) + \frac{\alpha}{\alpha + n} G_0(\boldsymbol{\theta}_{n+1}). \tag{2.11}$$

We can interpret the conditional distributions in terms of a simple urn model in which a ball of a distinct color is associated with each atom. The balls are drawn equiprobably; when a ball is drawn it is placed back in the urn together with another ball of the same color. In addition, with probability proportional to α a new atom is created by drawing from G_0 and a ball of a new color is added to the urn.

Equation (2.11) shows that $\boldsymbol{\theta}_{n+1}$ has positive probability of being equal to one of the previous draws. Moreover, there is a positive reinforcement effect; the more often a point is drawn, the more likely it is to be drawn in the future. To make the clustering property explicit, it is helpful to introduce a new set of variables that represent distinct values of the atoms. Define $\boldsymbol{\theta}^*_1, \ldots, \boldsymbol{\theta}^*_{N_C}$ to be the distinct values taken on by $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n$, and let m_k for $1 \leq k \leq N_C$ be the number of values that are equal to $\boldsymbol{\theta}_k$. We can re-express (2.11) as

$$\boldsymbol{\theta}_{n+1} \,|\, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n \sim \frac{n}{\alpha + n} \sum_{k=1}^{N_C} m_k \delta_{\boldsymbol{\theta}^*_k}(\boldsymbol{\theta}_{n+1}) + \frac{\alpha}{\alpha + n} G_0(\boldsymbol{\theta}_{n+1}).$$

Sometimes the Pólya urn scheme is described as the Chinese restaurant process (CRP). The name stems from the following process: Consider a Chinese restaurant with an unbounded number of tables (corresponding to the $\boldsymbol{\theta}^*_k$ in the DP). The n-th customer (corresponding to $\boldsymbol{\theta}_n$ in the DP) sits at the table indexed by $\boldsymbol{\theta}^*_k$ with probability proportional to the number m_k of guests already seated at this table, i.e. $\boldsymbol{\theta}_n = \boldsymbol{\theta}^*_k$. With probability proportional to α he sits at a table not yet occupied.

2.6.3 Dirichlet process mixture model

One of the most important applications of the Dirichlet process is as a *nonparametric prior* on the parameters of a mixture model. It allows us to describe mixture models where the number of mixtures isn't predefined by the researcher, but can float and adapt to the problem at hands; however the selection isn't completely automatically, as it's controlled by the parameter α. Suppose that observations $\boldsymbol{x}_n$ arise as follows:

$$\begin{aligned} G &\sim \mathrm{DP}(\alpha, G_0) \\ \boldsymbol{\theta}_n \,|\, G &\sim G(\boldsymbol{\theta}_n) \\ \boldsymbol{x}_n \,|\, \boldsymbol{\theta}_n &\sim F(\boldsymbol{x}_n | \boldsymbol{\theta}_n). \end{aligned} \tag{2.12}$$

This model is referred to as the *Dirichlet process mixture model* (MDP) [Antoniak, 1974] and is depicted as a graphical model in Figure 2.7. As can be seen from the graphical model, the factors $\boldsymbol{\theta}_n$ are conditionally independent given G, and the observation $\boldsymbol{x}_n$ is conditionally independent of the other observations given the factor $\boldsymbol{\theta}_n$.

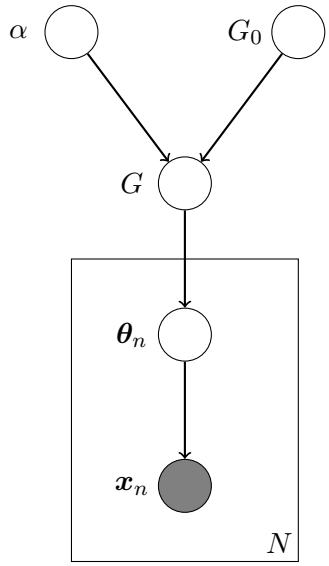


Figure 2.7: The Dirichlet process mixture model as a directed graphical model.

The stick-breaking representation can also be used for the Dirichlet process mixture, by adding an indicator variable z_i for each observation, which links an observation $\boldsymbol{x}_n$ with a factor $\boldsymbol{\theta}_k^*$ and is distributed according to $\boldsymbol{\pi}$. Let z_n be an assignment variable of the factor $\boldsymbol{\theta}_k^*$ with which the data point $\boldsymbol{x}_n$ is associated. The data $\boldsymbol{x}_{1:N}$ can be described as arising from the following process:

1. Draw $v_k \,|\, \alpha \sim \text{Beta}(1, \alpha), \quad k = 1, 2, \ldots$
2. Draw $\boldsymbol{\theta}_k^* \,|\, \boldsymbol{\lambda} \sim G_0(\boldsymbol{\lambda}), \quad k = 1, 2, \ldots$
3. For the n-th data point:
   a) Draw $z_n \,|\, \boldsymbol{\pi}(\boldsymbol{v}) \sim \text{Mult}(z_n|\boldsymbol{\pi}(\boldsymbol{v}), 1)$.
   b) Draw $\boldsymbol{x}_n \,|\, z_n \sim F(\boldsymbol{x}_n|\boldsymbol{\theta}_{z_n}^*)$.

The distribution of $\boldsymbol{x}_n$ conditional on z_n and $\{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty}$ is

$$p(\boldsymbol{x}_n|z_n, \{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty}) = \prod_{k=1}^{\infty} \left(h(\boldsymbol{x}_n) \exp\{\langle \boldsymbol{\theta}_k^*, \boldsymbol{x}_n \rangle - a(\boldsymbol{\theta}_k^*)\} \right)^{\mathbf{1}[z_n=k]}$$

where $a(\boldsymbol{\theta}_k^*)$ is the appropriate cumulant function and we assume for simplicity that $\boldsymbol{x}_n$ is the sufficient statistic for the natural parameter $\boldsymbol{\theta}^*$, i.e. $\boldsymbol{x}_n = \boldsymbol{s}(\boldsymbol{x}_n)$. Here we assumed that the likelihood is a member of the exponential family.

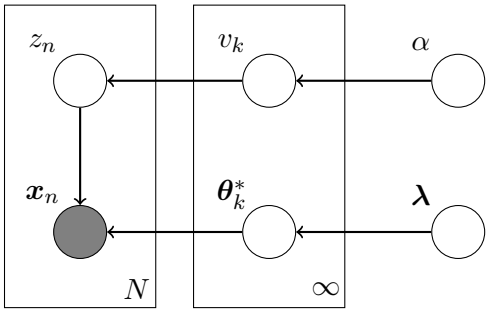


Figure 2.8: Graphical model representation of an exponential family Dirichlet process mixture in the stick-breaking construction.

2.6.4 The infinite limit of finite mixture models

A Dirichlet process mixture model can be derived as the limit of a sequence of finite mixture models, where the number of mixture components is taken to infinity.

$$\begin{aligned} \boldsymbol{\pi} &\sim \mathrm{Dir}(\alpha/N_C, \ldots, \alpha/N_C) \\ \boldsymbol{\theta}_k^* &\sim G_0(\boldsymbol{\lambda}) \\ z_n \,|\, \boldsymbol{\pi} &\sim \mathrm{Mult}(z_n|\boldsymbol{\pi}, 1) \\ \boldsymbol{x}_n \,|\, z_n, \{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C} &\sim F(\boldsymbol{x}_n|\boldsymbol{\theta}_{z_n}^*). \end{aligned} \tag{2.13}$$

By integrating over the mixing proportions, π_k, we can write the prior for z_{n+1} as the conditional probabilities of the following form:

$$P(z_{n+1} = k|z_1, \ldots, z_n) = \frac{m_{n,k} + \alpha/N_C}{n + \alpha}. \tag{2.14}$$

2.6.5 Expected number of clusters

In this section we give some insights into the expected number of clusters for a Dirichlet process. Formally, we are interested in $\mathrm{E}[N_C|N, \alpha]$, where N_C is the number of clusters, N is the number of samples and α is the concentration parameter of the Dirichlet process. We use the definition of the expectation and the fact that we create at outmost N clusters:

$$\mathrm{E}[N_C|N, \alpha] = \sum_{k=1}^{N} P(N_C = k|N, \alpha)k.$$

The probability $P(N_C = k|N, \alpha)$ can then be computed by recursion:

$$\begin{aligned} P(N_C = k|N, \alpha) = P(N_C = k|N-1, \alpha)\frac{N-1}{N-1+\alpha} \\ +P(N_C = k-1|N-1, \alpha)\frac{\alpha}{N-1+\alpha}. \end{aligned}$$

with

$$P(N_C = 1|N, \alpha) = \frac{1}{\alpha + 1} \cdots \frac{N - 1}{\alpha + N - 1} = \frac{\Gamma(\alpha + 1)\Gamma(N)}{\Gamma(\alpha + N)}$$

and $P(N_C = k|N, \alpha) = 0$ for $k > N$. In Figure 2.9 we show the probability distribution function for smaller values of N.

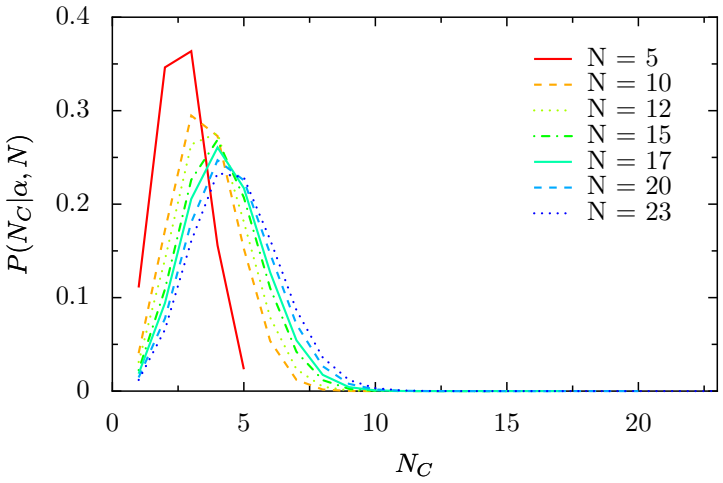


Figure 2.9: Probability distribution function of the number of clusters for $\alpha = 1.5$.

In Figure 2.10 we show the expected number of clusters for increasing N.

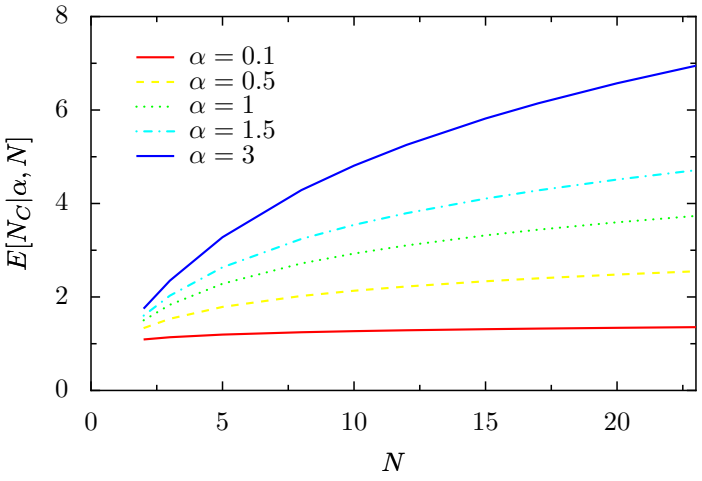


Figure 2.10: Expected number of clusters for increasing size of the data set.

In [West, 1992] one can find more results about the concentration parameter α of the DP and its relation to the number of clusters. One particularly interesting result, is that $N_C = \mathcal{O}(\ln N)$, i.e. the number of clusters grows in asymptotics logarithmically with the number of samples.

3 Inference for the Dirichlet Process

> I was just guessing. At numbers and figures.
> Pulling your puzzles apart.
>
> *(Coldplay in 'The Scientist')*

One can divide probabilistic inference algorithms for the Dirichlet Process broadly into two classes: sampling methods and variational optimization algorithms. Many of these methods have their roots in the statistical physics community and were for example originally used for finding solutions of the Ising model. As with most inference tasks, inference for the Dirichlet process mixture model is computationally expensive and the research focus has recently turned to finding efficient approximation algorithms. While the sampling algorithms have the theoretically appealing property that if run for an infinite amount of time, one would get the exact solution; variational methods generally don't have this property. However practically there's no big difference as the sampling algorithms are stopped after a certain number of iterations, and thus won't neither converge to the exact solution. The variational algorithms are especially promising, as they run in general much faster than sampling approaches. In this chapter we recapitulate some of the most important algorithms in use today.

3.1 Gibbs Sampling when Conjugate Priors are used

Most of this section is bluntly copied from the excellent article [Neal, 1998], as I felt there is no way to surpass the concise explanations.

The most direct approach to sampling for model (2.12) is to repeatedly draw values for each $\boldsymbol{\theta}_n$ from its conditional distribution given both the data and the $\boldsymbol{\theta}_i$ for $i \neq n$ (written as $\boldsymbol{\theta}_{-n}$). This conditional is obtained by combining the likelihood, written $F(\boldsymbol{x}_n|\boldsymbol{\theta}_n)$, and the prior conditional on $\boldsymbol{\theta}_{-n}$, which is

$$\boldsymbol{\theta}_n \,|\, \boldsymbol{\theta}_{-n} \sim \frac{1}{N-1+\alpha} \sum_{i \neq n} \delta_{\boldsymbol{\theta}_i}(\boldsymbol{\theta}_n) + \frac{\alpha}{N-1+\alpha} G_0(\boldsymbol{\theta}_n).$$

This can be derived from (2.11) by imaging that n is the last observation, as we may, since the observations are exchangeable. When combined with the likelihood, this yields the following conditional distribution for use in Gibbs sampling:

$$\boldsymbol{\theta}_n \,|\, \boldsymbol{\theta}_{-n}, \boldsymbol{x}_n \sim \sum_{i \neq n} q_{n,i} \delta_{\boldsymbol{\theta}_i}(\boldsymbol{\theta}_n) + r_n p(\boldsymbol{\theta}_n|\boldsymbol{x}_n). \tag{3.1}$$

Here, $p(\boldsymbol{\theta}_n|\boldsymbol{x}_n)$ is the posterior distribution for $\boldsymbol{\theta}_n$ based on the prior $G_0(\boldsymbol{\theta}_n)$ and the single observation $\boldsymbol{x}_n$, with likelihood $F(\boldsymbol{x}_n|\boldsymbol{\theta})$. The values of the $q_{n,i}$ and of r_n are defined as

$$\begin{aligned} q_{n,i} &= bF(\boldsymbol{x}_n|\boldsymbol{\theta}_i) \\ r_n &= b\alpha \int_{\Omega_{\boldsymbol{\theta}}} F(\boldsymbol{x}_n|\boldsymbol{\theta}) G_0(\boldsymbol{\theta})\ d\boldsymbol{\theta}\ , \end{aligned}$$

where b is such that $\sum_{i \neq n} q_{n,i} + r_n = 1$. For this Gibbs sampling method to be feasible, computing the integral defining r_n and sampling from $p(\boldsymbol{\theta}_n|\boldsymbol{x}_n)$ must be feasible operations. This will generally be so when G_0 is the conjugate prior for the likelihood given by F. We summarize this method in Algorithm 3.1.

Algorithm 3.1: Single assignment Gibbs sampler.

```
Let the state of the Markov chain consist of θ_1, ..., θ_N;
while not converged do
    for n = 1, ..., N do
        Draw a new value from θ_n | θ_{-n}, x_n as defined by equation (3.1);
    end
end
```

This algorithm is used by Escobar [1994] and Escobar and West [1995]. It produces an ergodic Markov chain, but convergence to the posterior distribution may be rather slow, and sampling thereafter may be inefficient. The problem is that there are often groups of observations that with high probability are associated with the same $\boldsymbol{\theta}$. Since the algorithm cannot change the $\boldsymbol{\theta}$ for more than one observation simultaneously, changes to the $\boldsymbol{\theta}$ values for observations in such a group can occur only rarely, as they require passage through a low-probability intermediate state in which observations in the group do not all have the same $\boldsymbol{\theta}$ value.

This problem is avoided if Gibbs sampling is instead applied to the model formulated as in (2.13), with the mixing proportions, π_k, integrated out. When N_C is finite, each Gibbs sampling scan consists of picking a new value for each z_n from its conditional distribution given $\boldsymbol{x}_n$, the $\{\boldsymbol{\theta}^*_k\}_{k=1}^{N_C}$, and the z_i for $i \neq n$ (written as z_{-n}), and then picking a new value for each $\boldsymbol{\theta}^*_k$ from its conditional distribution given the $\boldsymbol{x}_n$ for which $z_n = k$. The required conditional probabilities for z_n can easily be computed:

$$P(z_n = k|z_{-n}, \boldsymbol{x}_n, \boldsymbol{\theta}^*) = bF(\boldsymbol{x}_n|\boldsymbol{\theta}^*_k)\frac{m_{-n,k} + \alpha/N_C}{N - 1 + \alpha}\ ,$$

where $m_{-n,k}$ is the number of z_i for $i \neq n$ that are equal to k, and b is the appropriate normalizing constant. The last factor is derived from (2.14) by imaging that n is the last observation. (Note that the denominator $N - 1 + \alpha$ could be absorbed into b, but here and later it is retained for clarity.) The conditional distribution for $\boldsymbol{\theta}^*_k$ will also

be easy to sample from when the priors used are conjugate, and even when Gibbs sampling for $\boldsymbol{\theta}_k^*$ is difficult, one may simply substitute some other update that leaves the required distribution invariant. Note that when a new value is chosen for $\boldsymbol{\theta}^*$, the values of $\boldsymbol{\theta}_n = \boldsymbol{\theta}_{z_n}^*$ will change simultaneously for all observations associated with component k.

When N_C goes to infinity, we cannot, of course, explicitly represent the infinite number of $\boldsymbol{\theta}_k^*$. We instead represent, and do Gibbs sampling for, only those $\boldsymbol{\theta}_k^*$ that are currently associated with some observation. Gibbs sampling for the z_n is based on the following conditional probabilities (with $\boldsymbol{\theta}^*$ here being the set of $\boldsymbol{\theta}_k^*$ currently associated with at least one observation):

$$\begin{aligned} \text{If } k = z_i \text{ for some } i \neq n : P(z_n = k | z_{-n}, \boldsymbol{x}_n, \boldsymbol{\theta}^*) &= b \frac{m_{-n,k}}{N - 1 + \alpha} F(\boldsymbol{x}_n | \boldsymbol{\theta}_k^*) \\ P(z_n \neq z_i \text{ for all } i \neq n | z_{-n}, \boldsymbol{x}_n, \boldsymbol{\theta}^*) &= b \frac{\alpha}{N - 1 + \alpha} \int_{\Omega_{\boldsymbol{\theta}^*}} F(\boldsymbol{x}_n | \boldsymbol{\theta}^*) G_0(\boldsymbol{\theta}^*) \, d\boldsymbol{\theta}^* \end{aligned} \tag{3.2}$$

Here, b is the appropriate normalizing constant that makes the above probabilities sum to one. The numerical values of the z_i are arbitrary, as long as they faithfully represent whether or not $z_n = z_i$; they may be chosen for programming convenience, or to facilitate the display of mixture components in some desired order. When Gibbs sampling for z_n chooses a value not equal to any other z_i, a value for $\boldsymbol{\theta}_k^*$ is chosen from $p(\boldsymbol{\theta}^* | \boldsymbol{x}_n)$, the posterior distribution of $\boldsymbol{\theta}^*$ based on the prior G_0 and the single observation $\boldsymbol{x}_n$. We summarize this second Gibbs sampling method in Algorithm 3.2.

Algorithm 3.2: Simultaneous Gibbs sampler.

```
Let the state of the Markov chain consist of z_1, ..., z_N and θ*_1, ..., θ*_{N_C};
while not converged do
    for n = 1, ..., N do
        if m_{-n,z_n} = 0 then
            remove θ*_{z_n} from the state;
        end
        Draw a new value for z_n from z_n | z_{-n}, x_n, θ* as defined by equation (3.2);
        if z_n not associated with any other observation then
            Draw a value from p(θ*|x_n) and add it to the state;
        end
    end
    forall k ∈ {z_1, ..., z_N} do
        Draw a new value from θ*_k | x_n s.t. z_n = k;
    end
end
```

This is essentially the method used by Bush and MacEachern [1996] and by West et al. [1994]. As was the case for the first Gibbs sampling method, this approach is feasible if

we can compute $\int_{\Omega_{\boldsymbol{\theta}^*}} F(\boldsymbol{x}_n|\boldsymbol{\theta}^*)G_0(\boldsymbol{\theta}^*)\ d\boldsymbol{\theta}^*$ and sample from $p(\boldsymbol{\theta}^*|\boldsymbol{x}_n)$, as will generally be the case when G_0 is the conjugate prior.

Finally, in a conjugate context, we can often integrate analytically over the $\boldsymbol{\theta}_k^*$, eliminating them from the algorithm. The state of the Markov chain then consists only of the z_n, which we update by Gibbs sampling using the following conditional probabilities:

$$\begin{aligned} \text{If } k = z_i \text{ for some } i \neq n : P(z_n = k|z_{-n}, \boldsymbol{x}_n) &= b\frac{m_{-n,k}}{N-1+\alpha}\int_{\Omega_{\boldsymbol{\theta}^*}} F(\boldsymbol{x}_n|\boldsymbol{\theta}^*)p_{-n,k}(\boldsymbol{\theta}^*)\ d\boldsymbol{\theta}^* \\ P(z_n \neq z_i \text{ for all } i \neq j|z_{-n}, \boldsymbol{x}_n) &= b\frac{\alpha}{N-1+\alpha}\int_{\Omega_{\boldsymbol{\theta}^*}} F(\boldsymbol{x}_n|\boldsymbol{\theta}^*)G_0(\boldsymbol{\theta}^*)\ d\boldsymbol{\theta}^* \end{aligned} \tag{3.3}$$

Here, $p_{-n,k}$ is the posterior distribution of $\boldsymbol{\theta}^*$ based on the prior G_0 and all observations $\boldsymbol{x}_i$ for which $i \neq n$ and $z_n = k$. This third Gibbs sampling method is summarized in Algorithm 3.3.

Algorithm 3.3: Collapsed Gibbs sampler.

```
Let the state of the Markov chain consist of z_1, ..., z_N;
while not converged do
    for n = 1, ..., N do
        Draw a new value for z_n from z_n | z_{-n}, x_n as defined by equation (3.3);
    end
end
```

This algorithm is presented by MacEachern [1994] for mixtures of Gaussians and by Neal [1991] for models of categorical data.

3.2 Variational Inference

For more than a decade sampling based algorithms have been the major cornerstone for approximate inference in graphical models involving a Dirichlet process. However, recently researchers introduced promising variational methods which vastly outperform traditional Monte-Carlo algorithms in terms of running time on large-scale problems. Variational inference methods come at a price: they are usually harder to implement and might get stuck in local minima forever (and can thus in theory be quite poor approximations). The algorithm described here was first introduced by Blei and Jordan [2005]. More recently researchers pointed out some deficiencies in the design and proposed improved methods [Kurihara et al., 2007b,c], which they found to be also more efficient.

The algorithm of Blei and Jordan [2005] is based on the stick-breaking construction of the DP mixture (see subsection 2.6.1). As for all variational methods we use the bound on the log marginal probability introduced in subsection 2.5.2. The latent variables in the stick breaking construction are the stick lengths, the cluster parameters

and the cluster assignments: $\boldsymbol{W} = \{\boldsymbol{v}, \boldsymbol{\Theta}, \boldsymbol{z}\}$, where $\boldsymbol{v} = \{v_k\}_{k=1}^{\infty}$, $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty}$ and $\boldsymbol{z} = \{z_n\}_{n=1}^{N}$. The hyperparameters are the concentration parameter of the DP and the natural parameter of the conjugate base distribution: $\boldsymbol{\vartheta} = \{\alpha, \boldsymbol{\lambda}\}$.

Using the general equation (2.7) which we repeat here for clarity,

$$\ln p(\boldsymbol{X}|\boldsymbol{\vartheta}) \geq \mathrm{E}_q[\ln p(\boldsymbol{W}, \boldsymbol{X}|\boldsymbol{\vartheta})] - \mathrm{E}_q[\ln q_{\boldsymbol{\nu}}(\boldsymbol{W})],$$

we can give a bound on the log marginal probability of the data, for a variational distribution q:

$$\begin{aligned} \ln p(\boldsymbol{X}|\alpha, \boldsymbol{\lambda}) \geq\ & \mathrm{E}_q[\ln p(\boldsymbol{v}|\alpha)] + \mathrm{E}_q[\ln p(\boldsymbol{\Theta}|\boldsymbol{\lambda})] \\ & + \sum_{n=1}^{N} (\mathrm{E}_q[\ln p(z_n|\boldsymbol{v})] + \mathrm{E}_q[\ln p(\boldsymbol{x}_n|z_n, \boldsymbol{\Theta})]) \\ & - \mathrm{E}_q[\ln q(\boldsymbol{v}, \boldsymbol{\Theta}, \boldsymbol{z})]. \end{aligned} \tag{3.4}$$

To exploit this bound one now needs to specify the variational distribution q which approximates the distribution of the infinite-dimensional random measure G, which is given by the infinite sets $\{v_k\}_{k=1}^{\infty}$ and $\{\boldsymbol{\theta}_k^*\}_{k=1}^{\infty}$. Blei and Jordan consider the truncated stick-breaking representation, where instead of having an infinite number of sticks, one fixes a value T and let $q(v_T = 1) = 1$; this implies that the mixture proportions $\pi_k(\boldsymbol{v})$ are equal to zero for $k > T$. They thus introduce the following family of variational distributions for mean-field variational inference:

$$q(\boldsymbol{v}, \boldsymbol{\Theta}, \boldsymbol{z}) = \prod_{k=1}^{T-1} q_{\boldsymbol{\gamma}_k}(v_k) \prod_{k=1}^{T} q_{\boldsymbol{\tau}_k}(\boldsymbol{\theta}_k^*) \prod_{n=1}^{N} q_{\boldsymbol{\phi}_n}(z_n)$$

Where (as can be seen from the graphical model in Figure 2.8) $q_{\boldsymbol{\gamma}_k}(v_k)$ are Beta distributions with parameters $\gamma_{k,1}$ and $\gamma_{k,2}$, $q_{\boldsymbol{\tau}_k}(\boldsymbol{\theta}_k^*)$ are exponential family distributions with natural parameters $\boldsymbol{\tau}_k$ corresponding to the base measurement of the DP, and $q_{\boldsymbol{\phi}_n}(z_n)$ are Multinomial distributions for T bins with probabilities $\{\phi_{n,k}\}_{k=1}^{T}$. In the notation of section 2.5.2, the free variational parameters are

$$\boldsymbol{\nu} = \{\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_{T-1}, \boldsymbol{\tau}_1, \ldots, \boldsymbol{\tau}_T, \boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_N\}. \tag{3.5}$$

Below we now compute all the expectations for a DP for Dirichlet/Multinomial and Gaussian/Gaussian. These computations might be useful for other researchers trying to implement variational methods and being somewhat puzzled by the many expectations (as I was on the first sight). Others, who are not interested in a practical implementation of the variational algorithm might want to skip it, as it is quite technical and doesn't lead to many insights; however the first expectation is non-standard and it might thus be interesting to have a look at it.

3.2.1 Bound on the log marginal probability of the data

In this subsection we simplify all the expectations of the form $\mathrm{E}_q[\cdot]$ in equation (3.4), this will result in four computations needed to be carried out for each choice of prior and likelihood separately: $\mathrm{E}_q[a(\boldsymbol{\theta}_k^*)]$, $\mathrm{E}_q[\boldsymbol{\theta}_k^*]$, $Z_1(\boldsymbol{\lambda}, \chi)$ and $Z_2(\boldsymbol{x})$; this is done for Dirichlet/Multinomial and Gaussian/Gaussian. It should be pointed out, that most of these computations are not really needed for optimizing the bound, but they allow us to keep track of the progress.

$$\begin{aligned}\mathrm{E}_q[\ln p(z_n|\boldsymbol{v})] &= \mathrm{E}_q\left[\ln\left(\prod_{k=1}^{\infty}(1-v_k)^{\mathbf{1}[z_n>k]}v_k^{\mathbf{1}[z_n=k]}\right)\right] \\ &= \sum_{k=1}^{\infty} q(z_n>k)\mathrm{E}_q[\ln(1-v_k)] + q(z_n=k)\mathrm{E}_q[\ln v_k].\end{aligned}$$

Recall that $\mathrm{E}_q[\ln(1-v_T)] = 0$ and $q(z_n > T) = 0$. Consequently, we can truncate this summation at $k = T$:

$$\mathrm{E}_q[\ln p(z_n|\boldsymbol{v})] = \sum_{k=1}^{T} q(z_n>k)\mathrm{E}_q[\ln(1-v_k)] + q(z_n=k)\mathrm{E}_q[\ln v_k],$$

where

$$\begin{aligned}q(z_n=k) &= \phi_{n,k} \\ q(z_n>k) &= \sum_{i=k+1}^{T}\phi_{n,i} \\ \mathrm{E}_q[\ln v_k] &= \Psi(\gamma_{k,1}) - \Psi(\gamma_{k,1}+\gamma_{k,2}) \\ \mathrm{E}_q[\ln(1-v_k)] &= \Psi(\gamma_{k,2}) - \Psi(\gamma_{k,1}+\gamma_{k,2}).\end{aligned}$$

The digamma function, denoted by Ψ, arises from the derivative of the log normalization factor in the Beta distribution.

Furthermore, we need to compute the following expectations

$$\begin{aligned}\mathrm{E}_q[\ln p(\boldsymbol{v}|\alpha)] &= \sum_{k=1}^{T}\mathrm{E}_q\left[\ln\left(\frac{\Gamma(1+\alpha)}{\Gamma(1)\Gamma(\alpha)}v_k^0(1-v_k)^{\alpha-1}\right)\right] \\ &= \sum_{k=1}^{T}\mathrm{E}_q\left[\ln\left(\frac{\Gamma(1+\alpha)}{\Gamma(\alpha)}\right) + (\alpha-1)\ln(1-v_k)\right] \\ &= \sum_{k=1}^{T}\Big(\ln(\alpha) + (\alpha-1)\mathrm{E}_q[\ln(1-v_k)]\Big) \\ &= \sum_{k=1}^{T}\Big(\ln(\alpha) + (\alpha-1)(\Psi(\gamma_{k,2}) - \Psi(\gamma_{k,1}+\gamma_{k,2}))\Big)\end{aligned}$$

$$\mathrm{E}_q[\ln q(\boldsymbol{v}, \boldsymbol{\Theta}, \boldsymbol{z})] = \sum_{k=1}^{T-1} \mathrm{E}_q[\ln q_{\boldsymbol{\gamma}_k}(v_k)] + \sum_{k=1}^{T} \mathrm{E}_q[\ln q_{\boldsymbol{\tau}_k}(\boldsymbol{\theta}_k^*)] + \sum_{n=1}^{N} \mathrm{E}_q[\ln q_{\boldsymbol{\phi}_n}(z_n)]$$

$$\begin{aligned}
&\sum_{k=1}^{T-1} \mathrm{E}_q[\ln q_{\boldsymbol{\gamma}_k}(v_k)] \\
&\quad = \sum_{k=1}^{T-1} \mathrm{E}_q\left[\ln\left(\frac{\Gamma(\gamma_{k,1}+\gamma_{k,2})}{\Gamma(\gamma_{k,1})\Gamma(\gamma_{k,2})} v_k^{\gamma_{k,1}-1}(1-v_k)^{\gamma_{k,2}-1}\right)\right] \\
&\quad = \sum_{k=1}^{T-1}\left(\ln\left(\frac{\Gamma(\gamma_{k,1}+\gamma_{k,2})}{\Gamma(\gamma_{k,1})\Gamma(\gamma_{k,2})}\right) + (\gamma_{k,1}-1)\mathrm{E}_q[\ln(v_k)] + (\gamma_{k,2}-1)\mathrm{E}_q[\ln(1-v_k)]\right)
\end{aligned}$$

$$\begin{aligned}
\sum_{k=1}^{T} \mathrm{E}_q[\ln q_{\boldsymbol{\tau}_k}(\boldsymbol{\theta}_k^*)] &= \sum_{k=1}^{T} \mathrm{E}_q\left[\ln\left(\frac{1}{Z_1(\boldsymbol{\tau}_{k,1}, \tau_{k,2})} \exp(\langle \boldsymbol{\tau}_{k,1}, \boldsymbol{\theta}_k^* \rangle - \tau_{k,2} a(\boldsymbol{\theta}_k^*))\right)\right] \\
&= \sum_{k=1}^{T}\left(-\ln Z_1(\boldsymbol{\tau}_{k,1}, \tau_{k,2}) + \langle \boldsymbol{\tau}_{k,1}, \mathrm{E}_q[\boldsymbol{\theta}_k^*] \rangle - \tau_{k,2}\mathrm{E}_q[a(\boldsymbol{\theta}_k^*)]\right)
\end{aligned}$$

$$\begin{aligned}
\sum_{n=1}^{N} \mathrm{E}_q[\ln q_{\boldsymbol{\phi}_n}(z_n)] &= \sum_{n=1}^{N} \mathrm{E}_q\left[\ln\left(\prod_{k=1}^{T} \phi_{n,k}^{\mathbf{1}[z_n=k]}\right)\right] \\
&= \sum_{n=1}^{N}\sum_{k=1}^{T} q(z_n = k)\ln(\phi_{n,k}) \\
&= \sum_{n=1}^{N}\sum_{k=1}^{T} \phi_{n,k}\ln(\phi_{n,k})
\end{aligned}$$

$$\begin{aligned}
\sum_{k=1}^{T} \mathrm{E}_q[\ln p(\boldsymbol{\theta}_k^* | \boldsymbol{\lambda})] &= \sum_{k=1}^{T} \mathrm{E}_q\left[\ln\left(\frac{1}{Z_1(\boldsymbol{\lambda}, \chi)} \exp(\langle \boldsymbol{\lambda}, \boldsymbol{\theta}_k^* \rangle - \chi a(\boldsymbol{\theta}_k^*))\right)\right] \\
&= \sum_{k=1}^{T}\left(-\ln(Z_1(\boldsymbol{\lambda}, \chi)) + \langle \boldsymbol{\lambda}, \mathrm{E}_q[\boldsymbol{\theta}_k^*] \rangle - \chi\mathrm{E}_q[a(\boldsymbol{\theta}_k^*)]\right)
\end{aligned}$$

$$\sum_{n=1}^{N} \mathrm{E}_q[\ln p(\boldsymbol{x}_n | z_n, \boldsymbol{\Theta})] = \sum_{n=1}^{N}\sum_{k=1}^{T} q(z_n = k)(-\ln(Z_2(\boldsymbol{x}_n)) + \langle \mathrm{E}_q[\boldsymbol{\theta}_k^*], \boldsymbol{x}_n \rangle - \mathrm{E}_q[a(\boldsymbol{\theta}_k^*)])$$

For a specific choice of conjugate prior and likelihood one has thus to compute $Z_1(\boldsymbol{\lambda}, \chi)$, $Z_2(\boldsymbol{x})$, $\mathrm{E}_q[\boldsymbol{\theta}_k^*]$ and $\mathrm{E}_q[a(\boldsymbol{\theta}_k^*)]$. This is done below for two configurations: Dirichlet/Multinomial and Gaussian/Gaussian. All the other computations are independent of the prior and likelihood (provided they are conjugate and from the exponential family).

Dirichlet/Multinomial

The expectation of the log partition function is zero,

$$\mathrm{E}_q[a(\boldsymbol{\theta}_k^*)] = 0.$$

For the expectation of the factors we introduce $\beta_k = \sum_{j=1}^{K} \tau_{k,1,j} + K$ where K is the number of bins, the expectation is then given by

$$\mathrm{E}_q[\boldsymbol{\theta}_k^*] = [\Psi(\tau_{k,1,1} + 1) - \Psi(\beta_k), \ldots, \Psi(\tau_{k,1,K} + 1) - \Psi(\beta_k)]^T.$$

Furthermore, the two normalization functions are given by

$$Z_1(\boldsymbol{\lambda}, \chi) = \sum_{j=1}^{K} \ln(\Gamma(\lambda_j + 1)) - \ln \Gamma\left(\sum_{j=1}^{K} \lambda_j + K\right),$$

and

$$Z_2(\boldsymbol{x}) = \frac{x_1! \cdots x_K!}{(\sum_{j=1}^{K} x_j)!}.$$

Gaussian/Gaussian

See page 16 for more information about the notation. The expectation of the log partition function is zero, i.e.

$$\mathrm{E}_q[a(\boldsymbol{\theta}_k^*)] = 0,$$

as the log partition function is not present in equation (2.6). The expectation of the factors is given by

$$\mathrm{E}_q[\boldsymbol{\theta}_k^*] = \mathrm{E}\left[[\boldsymbol{\mu}, -\frac{1}{2}\boldsymbol{\mu}\boldsymbol{\mu}^T]\right] = [\boldsymbol{\mu}_\theta, -\frac{1}{2}(\boldsymbol{\Sigma}_\theta + \boldsymbol{\mu}_\theta \boldsymbol{\mu}_\theta^T)]$$

where $\boldsymbol{\Sigma}_\theta := \boldsymbol{\tau}_{k,1,2}^{-1}$ and $\boldsymbol{\mu}_\theta := \boldsymbol{\Sigma}_\theta \boldsymbol{\tau}_{k,1,1}$. Note that the χ factor cancels out in the computation of $\boldsymbol{\mu}_\theta$.

For computing the expectation $\mathrm{E}_q[\boldsymbol{\theta}_k^*]$ we use the fact that the expectation of a Gaussian is given by its mean, here denoted by $\boldsymbol{\mu}_\theta$, and $\mathrm{E}[\boldsymbol{\mu}\boldsymbol{\mu}^T]$ can be computed from $\boldsymbol{\Sigma}_\theta$ and $\boldsymbol{\mu}_\theta$ by

$$\begin{aligned}
\boldsymbol{\Sigma}_\theta &= \mathrm{E}[(\boldsymbol{\mu} - \boldsymbol{\mu}_\theta^T)(\boldsymbol{\mu} - \boldsymbol{\mu}_\theta)] \\
&= \mathrm{E}[\boldsymbol{\mu}\boldsymbol{\mu}^T - 2\boldsymbol{\mu}\boldsymbol{\mu}_\theta^T + \boldsymbol{\mu}_\theta \boldsymbol{\mu}_\theta^T] \\
&= \mathrm{E}[\boldsymbol{\mu}\boldsymbol{\mu}^T] - 2\mathrm{E}[\boldsymbol{\mu}]\boldsymbol{\mu}_\theta^T + \boldsymbol{\mu}_\theta \boldsymbol{\mu}_\theta^T
\end{aligned}$$

and thus $\mathrm{E}[\boldsymbol{\mu}\boldsymbol{\mu}^T] = \boldsymbol{\Sigma}_\theta + \boldsymbol{\mu}_\theta \boldsymbol{\mu}_\theta^T$.

The two normalization functions are given by

$$Z_1(\boldsymbol{\lambda}, \chi) = \frac{(2\pi)^{d/2} \det(\boldsymbol{\lambda}_2^{-1})^{1/2}}{\exp\left(-\frac{1}{2}\boldsymbol{\lambda}_1^T \boldsymbol{\lambda}_2^{-1} \boldsymbol{\lambda}_1\right)},$$

and

$$Z_2(\boldsymbol{x}) = \frac{(2\pi)^{d/2} \det(\boldsymbol{x}_2^{-1})^{1/2}}{\exp\left(-\frac{1}{2}\boldsymbol{x}_1^T \boldsymbol{x}_2^{-1} \boldsymbol{x}_1\right)}.$$

3.2.2 Coordinate ascent algorithm

Using the general expression in equation (2.8) we can derive a mean-field coordinate ascent algorithm. This yields:

$$\begin{aligned}
\gamma_{k,1} &= 1 + \sum_{n=1}^{N} \phi_{n,k} \\
\gamma_{k,2} &= \alpha + \sum_{n=1}^{N} \sum_{i=k+1}^{T} \phi_{n,i} \\
\boldsymbol{\tau}_{k,1} &= \boldsymbol{\lambda} + \sum_{n=1}^{N} \phi_{n,k} \boldsymbol{x}_n \\
\tau_{k,2} &= \chi + \sum_{n=1}^{N} \phi_{n,k} \\
\phi_{n,k} &\propto \exp(S_{n,k}),
\end{aligned}$$

for $k \in \{1, \ldots, T\}$ and $n \in \{1, \ldots, N\}$, where

$$S_{n,k} = \mathrm{E}_q[\ln v_k] + \sum_{j=1}^{k-1} \mathrm{E}_q[\ln(1 - v_j)] + \mathrm{E}_q[\boldsymbol{\theta}_k^*]^T \boldsymbol{x}_n - \mathrm{E}_q[a(\boldsymbol{\theta}_k^*)].$$

Iterating these updates optimizes equation (3.4) with respect to the variational parameters defined in equation (3.5).

4 Comparative study of clustering methods

> You made up your mind to leave it all behind.
> Now you're forced to fight it out.
>
> *(The Fray in 'Fall Away')*

4.1 Theoretical comparison of the Dirichlet process mixture and information theoretic MOS criteria

In this section we show that there's theoretical evidence to believe that the Dirichlet process and the information theoretic approaches, especially the BIC, are not that much different after all. However, as we will see in section 4.2 this is only valid up to a certain extent.

DPM models select the model order by sampling from a posterior of the form

$$\frac{n}{\alpha+n}\sum_{i=1}^{n}\delta_{\boldsymbol{\theta}_i}(\boldsymbol{\theta})+\frac{\alpha}{\alpha+n}G_0(\boldsymbol{\theta})=:\frac{n}{\alpha+n}\hat{G}_n(\boldsymbol{\theta})+\frac{\alpha}{\alpha+n}G_0(\boldsymbol{\theta})$$

which is additionally convolved with an observed-data likelihood $F(\cdot)$. Now assume that, instead of sampling, we were to choose a component by maximization. That is, G_0 is selected (and the model split) if

$$n\int_{\Omega_{\boldsymbol{\theta}}}F(\boldsymbol{x}|\boldsymbol{\theta})\hat{G}_n(\boldsymbol{\theta})\;d\boldsymbol{\theta}<\alpha\int_{\Omega_{\boldsymbol{\theta}}}F(\boldsymbol{x}|\boldsymbol{\theta})G_0(\boldsymbol{\theta})\;d\boldsymbol{\theta} \tag{4.1}$$

The integral on the lhs is ($m_{n,k}$ denotes the number of samples assigned to cluster k)

$$\int_{\Omega_{\boldsymbol{\theta}}}F(\boldsymbol{x}|\boldsymbol{\theta})\hat{G}_n(\boldsymbol{\theta})\;d\boldsymbol{\theta}=\sum_{k=1}^{N_C}m_{n,k}F(\boldsymbol{x}|\boldsymbol{\theta}_k^*)=:g_n(\boldsymbol{x})$$

and hence, up to scaling, just the likelihood of $\boldsymbol{x}$ under the finite mixture model given by $\sum_{k=1}^{N_C}\frac{m_{n,k}}{n}F(\boldsymbol{x}|\boldsymbol{\theta}_k^*)$. Let's denote the base measure integral as

$$g_0(\boldsymbol{x}):=\int_{\Omega_{\boldsymbol{\theta}}}F(\boldsymbol{x}|\boldsymbol{\theta})G_0(\boldsymbol{\theta})\;d\boldsymbol{\theta}.$$

Now we can rewrite (4.1) to get

$$\begin{aligned}ng_n(\boldsymbol{x})&<\alpha g_0(\boldsymbol{x})\\ \frac{g_n(\boldsymbol{x})}{g_0(\boldsymbol{x})}&<\frac{\alpha}{n}.\end{aligned}$$

The numerator on the lhs is a likelihood for given $\boldsymbol{x}$, and the denominator can be regarded as the likelihood of $\boldsymbol{x}$ under a "world model" that mixes according to G_0. The lhs can thus be regarded as a likelihood ratio statistic testing "no split" against the "split" event.

Considering the following model order selection procedure: A finite mixture model

$$M_{N_C}(\boldsymbol{x}|\{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}, \boldsymbol{\pi}) := \sum_{k=1}^{N_C} \pi_k F(\boldsymbol{x}|\boldsymbol{\theta}_k^*)$$

has been estimated on n observations. A new observation $\boldsymbol{x}_{n+1}$ may either be explained by the current model, or by an additional component $F(\cdot\,|\boldsymbol{\theta}_{N_C+1}^*)$ with mixture weight $\pi_{N_C+1} = \frac{1}{n+1}$. The modified model is denoted by M_{N_C+1}. Assume first that $\boldsymbol{\theta}_{N_C+1}^*$ is predefined. To decide whether or not the observation $\boldsymbol{x}_{n+1}$ solicits a split, we can compare the models with a likelihood ratio test statistic. The model is split if the modified model achieves a higher likelihood score, i.e. if

$$\frac{M_{N_C}(\boldsymbol{x}_{n+1}|\{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}, \boldsymbol{\pi})}{M_{N_C+1}(\boldsymbol{x}_{n+1}|\{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C+1}, \boldsymbol{\pi})} < 1. \tag{4.2}$$

Instead of the plain likelihood test, we may opt for an information criterion based test: AIC/BIC scores are computed for both models, and the model with the smaller score is selected. For any model with likelihood $\ell(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$, the AIC score is

$$\mathrm{AIC}_{N_C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n|\{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}, \boldsymbol{\pi}) := -\ln \ell(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n|\{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}, \boldsymbol{\pi}) + \kappa(N_C),$$

where $\kappa(N_C)$ denotes the number of free model parameters. The BIC score is

$$\mathrm{BIC}_{N_C}(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n|\{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}, \boldsymbol{\pi}) := -\ln \ell(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n|\{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C}, \boldsymbol{\pi}) + \frac{1}{2}\kappa(N_C) \ln n.$$

These scores are standard and are usually used in a global, brute-force manner: We run a clustering algorithm on the complete data set for different number of clusters and compare the likelihood returned by them with the scores as given above. We choose the model with the smallest score. Here, however we will use the scores as a local criterion for splitting the model based on a single observation. We will omit the conditioning on the model to keep the notation uncluttered. The likelihood ratio test in (4.2) can be substituted by an AIC comparison (note that the log likelihood terms up to the data point $\boldsymbol{x}_{n+1}$ can be ignored, as they are the same on each side of the equation):

$$\begin{aligned}
\mathrm{AIC}_{N_C}(\boldsymbol{x}_{n+1}) &> \mathrm{AIC}_{N_C+1}(\boldsymbol{x}_{n+1}) \\
-\ln M_{N_C}(\boldsymbol{x}_{n+1}) + \kappa(N_C) &> -\ln M_{N_C+1}(\boldsymbol{x}_{n+1}) + \kappa(N_C+1) \\
\frac{M_{N_C}(\boldsymbol{x}_{n+1})}{M_{N_C+1}(\boldsymbol{x}_{n+1})} &< \exp\Big(\kappa(N_C) - \kappa(N_C+1)\Big).
\end{aligned}$$

Similarly, for BIC:

$$\begin{aligned}
\mathrm{BIC}_{N_C}(\boldsymbol{x}_{n+1}) &> \mathrm{BIC}_{N_C+1}(\boldsymbol{x}_{n+1}) \\
-\ln M_{N_C}(\boldsymbol{x}_{n+1}) + \frac{\kappa(N_C)}{2}\ln(n+1) &> -\ln M_{N_C+1}(\boldsymbol{x}_{n+1}) + \frac{\kappa(N_C+1)}{2}\ln(n+1) \\
\frac{M_{N_C}(\boldsymbol{x}_{n+1})}{M_{N_C+1}(\boldsymbol{x}_{n+1})} &< \frac{1}{n+1}\exp\left(\frac{1}{2}(\kappa(N_C)-\kappa(N_C+1))\right).
\end{aligned}$$

4.2 Splitting information criterion

The discussion in the previous section showed that the algorithms based on the information criterion are very similar under certain conditions to a Dirichlet process. Here we describe a new algorithm that is based on the results of the previous section; it combines ideas from the information theoretic models and the Dirichlet process. However, as we will see, this algorithm shows to have serious drawbacks, which we will discuss later on.

Algorithm 4.1: Fusion of the Dirichlet process and BIC/AIC

Let the state of the algorithm consist of $z_1, \ldots, z_N$, $\pi_1, \ldots, \pi_{N_C}$ and $\boldsymbol{\theta}^*_1, \ldots, \boldsymbol{\theta}^*_{N_C}$;
while *not converged* **do**
  for $n = 1, \ldots, N$ **do**
    if $m_{-n,z_n} = 0$ **then**
      remove $\boldsymbol{\theta}^*_{z_n}$ from the state;
      remove π_{z_n} from the state;
      set the π_k according to the $m_{-n,k}$;
    end
    if $S_{N_C}(\boldsymbol{x}_n) < S_{N_C+1}(\boldsymbol{x}_n)$ **then**
      $N_C = N_C + 1$;
      $z_n = N_C$;
      $\boldsymbol{\theta}^*_{N_C} = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\boldsymbol{x}_n)$ and add it to the state;
      add π_{N_C} to the state;
      set the π_k according to the $m_{n,k}$;
    else
      $z_n = \arg\max_k p(\boldsymbol{x}_n|\boldsymbol{\theta}^*_k)$;
      set the π_k according to the $m_{n,k}$;
    end
  end
end

Where S_{N_C} is either given by BIC or AIC and where we dropped the parameters, in favour for an uncluttered notation. $m_{n,k}$ denotes the number of samples assigned to cluster k when sample $\boldsymbol{x}_n$ is included, $m_{-n,k}$ has the same meaning, but with $\boldsymbol{x}_n$ excluded. We tested this algorithm on two simple data sets. The results are shown in Figure 4.1.

While the algorithm performs well, and very similar to the DP-Gibbs sampler for small covariance, the algorithm is considerably worse for larger data sets. Looking at the results, one observes that especially the BIC version of the algorithm shows unexpected behaviour: it gets worse for larger data sets. Can we explain this? Consider a data set with 3 clusters.

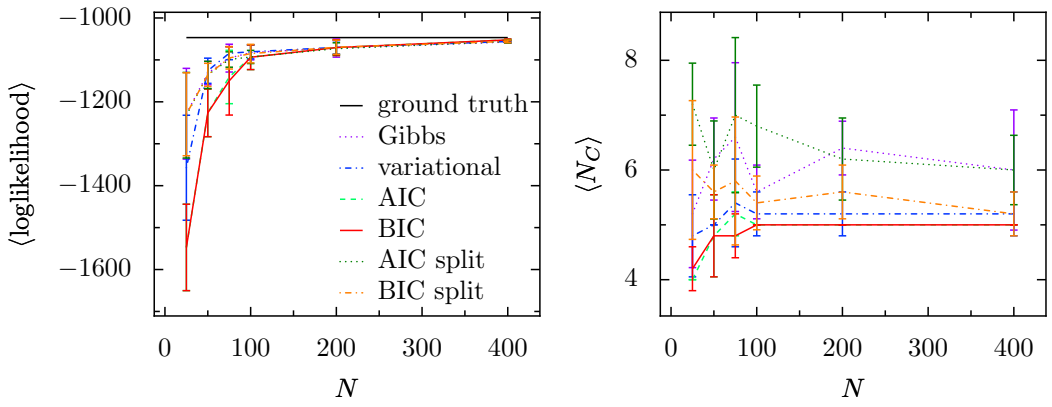
- In the standard BIC/AIC formulation we run in a brute-force manner, for all possible[1] values for N_C, an EM clustering algorithm and add the complexity term according to AIC/BIC to compare the different models. However, this complexity costs are "paid" by all the data points. Say, if the third cluster contains 10 data points, this third cluster is voted for by 10 data points. In the algorithm above, however, we will justify the introduction of a new cluster solely based on a single data point, which for large enough variance might not suffice to introduce the additional cluster. The problem could be summarized as a *global versus local MOS criterion.*

- However, the Dirchlet process, also decides only based on a single observation, whether we should add an additional cluster. Why should the DP not suffer from the same problem? The important difference here is that in the Dirichlet formulation we have a probabilistic decision, whereas in the algorithm above the decision is a maximization. To illustrate the difference assume that for the DP each of the 10 data points has individually a probability of 0.2 for introducing a new cluster. The probability, however, that at least one new cluster is created is approximately 0.9, which can be computed by the Binomial distribution. This means that in the DP formulation we are likely to create the third cluster, while in the algorithm above, this will never happen.

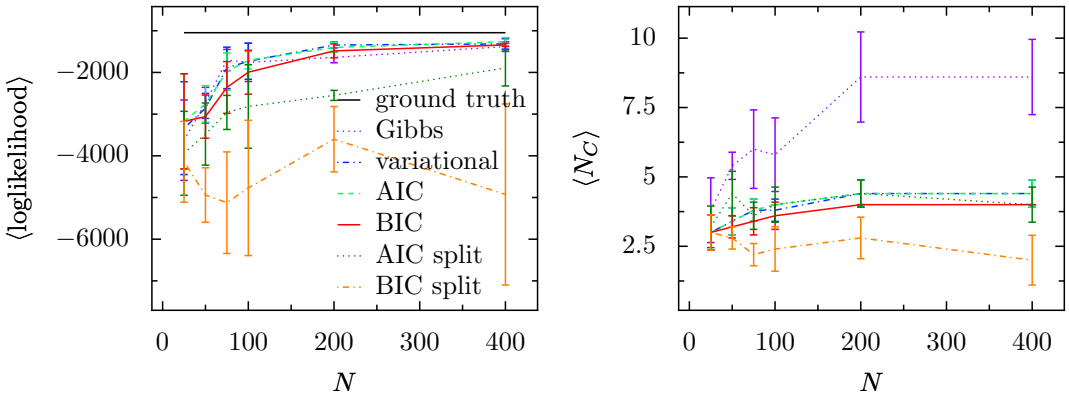
- The BIC formulation punishes models more aggresively than the AIC, especially for large data sets. It is thus to be expected, that the problem becomes even more evident for BIC. However, as can be seen in Figure 4.1, it also happens for the AIC splitting.

The informal discussion above should have shown that Algorithm 4.1 can not compete with the DP or the brute-force information theoretic algorithms and should not be used in practice. For this reason, we also won't include it in the comparison.

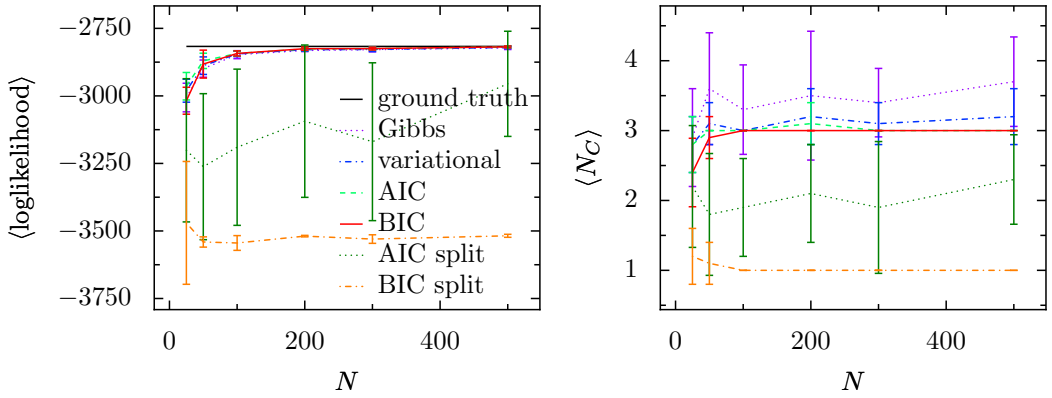
[1] in practice an upper bound on the number of clusters will be set.



(a) Gaussian data, $\sigma^2 = 0.1$



(b) Gaussian data, $\sigma^2 = 0.7$



(c) Multinomial data

Figure 4.1: The splitting algorithms compared to the brute-force algorithms. The latter work better, especially for harder configurations.

4.3 Evaluation

In the following we will always assume a mixture model is given and we generate samples from this model, which results in data of which we already know the cluster assignments, the components and the mixing proportions. We are then interested in how well the different clustering algorithms reconstruct these parameters, however, what's a good measure for this task?

There are different ways to evaluate the performance of a clustering method. For traditional clustering algorithms, where the number of components is assumed to be known, one usually uses the the log likelihood of the data. We will base our comparison of MOS strategies mostly on the log likelihood, too, as we found this to lead to sensible results. However, we did use a special validation data set for the comparison of the log likelihood. Beside the log likelihood, there exist also a number of alternative measures, which however all turned out to show suboptimal effects, we list them below.

expected number of clusters For the MOS quality one could compare the expected number of clusters with the true number of clusters. However this gives little indication of the quality of the model, as this measure doesn't consider the cluster factors nor the mixing proportions.

assignment error Compare the correct cluster assignment with the assignment inferred by the clustering algorithms, as the assignment is only known up to permutations, we first need to find a mapping of the inferred assignments to the given assignments. One can solve this for example in an optimistic way by using the Hungarian method [Kuhn, 1955], an algorithm from theoretical computer science. However this measure is not suitable for unbalanced datasets, as an algorithm might never identify a small cluster but still only have a relatively small assignment error.

Hubert's Γ index, Rand's index and Jaccard's index Similar in spirit, and all of them measuring quantities similar to the assignment error. For more details see the subsection at the end of this section.

As already mentioned we evaluated the different scores and measurements and found the log likelihood to best capture the goodness of fit, "overfitting" should be prevented by working on a validation set that is not used by the clustering algorithms. Also, while overfitting is for example possible in GMM, when we also infer the covariance of the clusters (each data point gets a Dirac impulse), we don't infer the variance here and thus overfitting can already not happen on the normal data.

4.3.1 Running time

As we were mainly interested in comparing the different approaches to model clusterings with automatic[2] selection of the number of clusters, we did not compare the running

[2]To some extent, the DP still has an concentration parameter α.

time of the different algorithms/models in an elaborate manner. We refrained from doing so, because such a comparison is highly implementation dependent; as most of the programming for this master's thesis was done in MATLAB, the variational algorithm has for example an advantage, as most of its operations can be formulated efficiently as matrix/vector computations, this is usually not the case for sampling algorithms, such as the Gibbs sampler.

As the running time is a very important criterion for an algorithm, we can still give an highly informal comparison of the algorithms: The variational algorithm is according to our observations non-surprisingly quite fast and has the above mentioned advantage that most of the operations can be described as vector/matrix manipulations. The collapsed Gibbs algorithm is the most accurate and fastest out of the three Gibbs algorithms described in section 3.1. However, according to our experiments, it is still considerably slower than the variational inference, especially for large data sets. This, again, comes as no surprise, as variational algorithms are usually considered preferable for large-scale problems, because of their efficiency. Our AIC and BIC implementations are faster than the Gibbs sampling, but still slower than the variational algorithm, especially the repeated clustering for different number of clusters makes the algorithm slow.

4.3.2 Hubert's Γ index

This subsection is taken from [Law et al., 2002]. Given the data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ and two clustering results $\mathcal{C}_1$ and $\mathcal{C}_2$ (with number of clusters $N_C^{(1)}$ and $N_C^{(2)}$ respectively), define

$$
\begin{aligned}
I_k(\boldsymbol{x}_i, \boldsymbol{x}_j) &= \begin{cases} 1 & \text{if } \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j \text{ are in the same cluster in } \mathcal{C}_k \\ 0 & \text{otherwise} \end{cases} \\
a &= \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} I_1(\boldsymbol{x}_i, \boldsymbol{x}_j) I_2(\boldsymbol{x}_i, \boldsymbol{x}_j) \\
b &= \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} I_1(\boldsymbol{x}_i, \boldsymbol{x}_j)(1 - I_2(\boldsymbol{x}_i, \boldsymbol{x}_j)) \\
c &= \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (1 - I_1(\boldsymbol{x}_i, \boldsymbol{x}_j)) I_2(\boldsymbol{x}_i, \boldsymbol{x}_j) \\
d &= \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (1 - I_1(\boldsymbol{x}_i, \boldsymbol{x}_j))(1 - I_2(\boldsymbol{x}_i, \boldsymbol{x}_j)).
\end{aligned}
$$

So a represents the number of pairs of data that are in the same cluster in both $\mathcal{C}_1$ and $\mathcal{C}_2$. b denotes the number of pairs of data that are in the same cluster in $\mathcal{C}_1$ but not in $\mathcal{C}_2$, while c denotes the number of pairs of data that are in the same cluster in $\mathcal{C}_2$ but not in $\mathcal{C}_1$. Finally, d means the number of pairs of data that are in different clusters in

$\mathcal{C}_1$ or $\mathcal{C}_2$. There is an efficient way to compute a, b, c an d. Define n_{ij} to be the number of data that are that both in the i-th cluster of $\mathcal{C}_1$ and the j-th cluster of $\mathcal{C}_2$. Let m_i be the total number of pairs of data that are in the same cluster in $\mathcal{C}_i$. Denote the total number of pairs of data by M. Then

$$
\begin{aligned}
M &= \frac{1}{2}N(N-1) \\
m_1 &= \frac{\sum_{i=1}^{N_C^{(1)}}(\sum_{j=1}^{N_C^{(2)}} n_{ij})(\sum_{j=1}^{N_C^{(2)}} n_{ij}-1)}{2} = \frac{1}{2}\Big(\sum_{i=1}^{N_C^{(1)}}(\sum_{j=1}^{N_C^{(2)}} n_{ij})^2 - N\Big) \\
m_2 &= \frac{\sum_{i=1}^{N_C^{(2)}}(\sum_{j=1}^{N_C^{(1)}} n_{ij})(\sum_{j=1}^{N_C^{(1)}} n_{ij}-1)}{2} = \frac{1}{2}\Big(\sum_{i=1}^{N_C^{(2)}}(\sum_{j=1}^{N_C^{(1)}} n_{ij})^2 - N\Big) \\
a &= \sum_{i=1}^{N_C^{(1)}}\sum_{j=1}^{N_C^{(2)}} \frac{n_{ij}(n_{ij}-1)}{2} = \frac{1}{2}\Big(\sum_{i=1}^{N_C^{(1)}}\sum_{j=1}^{N_C^{(2)}} n_{ij}^2 - N\Big) \\
b &= m_1 - a \\
c &= m_2 - a \\
d &= M - a - b - c
\end{aligned}
$$

The similarity measures between $\mathcal{C}_1$ and $\mathcal{C}_2$ are then given as follows:

- Rand's index:
$$(a+d)/M$$
- Jaccard's index:
$$a/(a+b+c)$$
- Hubert's Γ index
$$\frac{Ma - m_1 m_2}{\sqrt{m_1 m_2 (M-m_1)(M-m_2)}}$$

Note that the ranges of Rand's index and Jaccard's index are both $[0, 1]$. The larger these two indices, the more similar $\mathcal{C}_1$ and $\mathcal{C}_2$. For Hubert's Γ index, the range is $[-1, 1]$. $\mathcal{C}_1$ and $\mathcal{C}_2$ are regarded as more similar if the absolute value of the index is closer to one. Note that Hubert's Γ index is undefined when either $N_C^{(1)} = 1$ or $N_C^{(2)} = 1$.

4.4 Data from a Bayesian finite mixture model

In this section we compare the different clustering methods that we introduced in preceding chapters. We do this on synthetic data from a finite mixture model. It has to be pointed out, that this is not data from a Dirichlet process and it is thus interesting how clustering methods based on the Dirichlet process perform compared to more traditional methods. The data we use is, except for two data sets, fairly balanced in the sense that

the different clusters have around the same size, this is typically not the case for data from a Dirichlet process, there one observes smaller clusters.

For this experiment we fix the number of clusters, here denoted by N_C, and generate the data as follows:

$$\begin{aligned} \boldsymbol{\pi} &\sim \mathrm{Dir}(\beta\alpha/N_C, \ldots, \beta\alpha/N_C) \\ \boldsymbol{\theta}_k^* &\sim G_0(\boldsymbol{\lambda}) \quad 1 \leq k \leq N_C \\ z_n \,|\, \boldsymbol{\pi} &\sim \mathrm{Mult}(z_n|\boldsymbol{\pi}, 1) \\ \boldsymbol{x}_n \,|\, z_n, \{\boldsymbol{\theta}_k^*\}_{k=1}^{N_C} &\sim F(\boldsymbol{x}_n|\boldsymbol{\theta}_{z_n}^*). \end{aligned} \tag{4.3}$$

With $\beta = 1$ this is the same as in section 2.6.4: a Dirichlet process in the limit when $N_C \to \infty$, however we used a finite value for N_C and also set β to values much larger than 1; this results in well balanced clusters, as the components of $\boldsymbol{\pi}$ are close to $1/N_C$. We generate the results for different configurations sampled from the process described in equation (4.3), additionally we alter the number of clusters N_C, too.

4.4.1 Gaussian data

Zurich dataset: 3 clusters

The Zurich dataset characteristics and our experiments are summarized in Table 4.1.

type	Gaussian: $\boldsymbol{\mu}_\theta = [0, 0], \boldsymbol{\Sigma}_{\boldsymbol{\theta}} = 3\boldsymbol{I}, \boldsymbol{\Sigma}_{\boldsymbol{x}} = \sigma^2\boldsymbol{I}$
dimension	2
number of clusters N_C	3
balanced	yes
σ^2	$0.05, 0.1, 0.2, 0.3, 0.4$
α	0.2
N	$25, 50, 100, 200, 300, 500$
repetitions	10
algorithms run	collapsed Gibbs, variational, AIC, BIC

Table 4.1: The Zurich dataset and the experiments we performed on it.

We show samples from the two different configurations of the Zurich dataset in Figure 4.2.

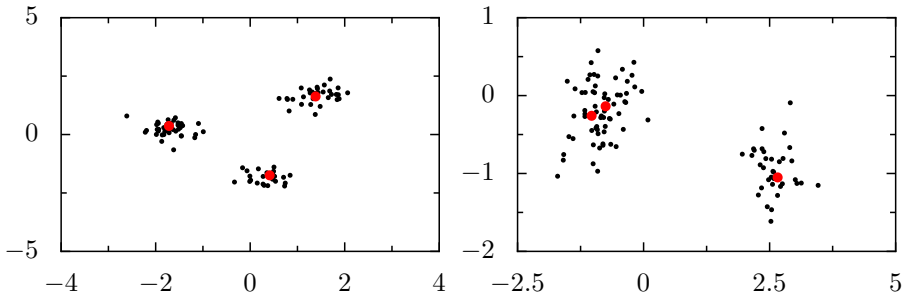
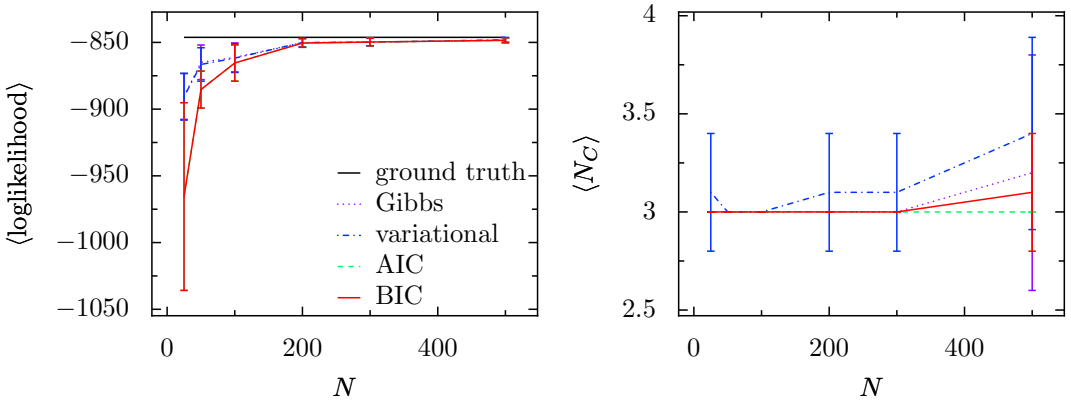
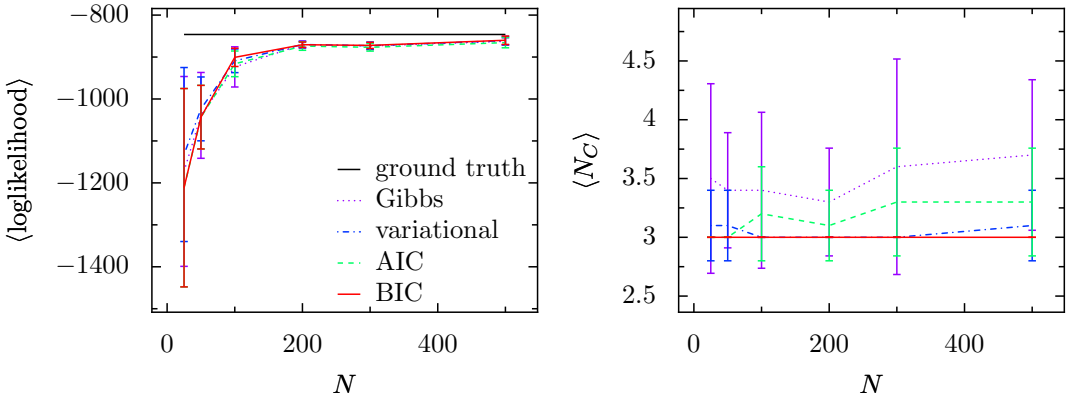


Figure 4.2: On the left we see samples from the easier Zurich dataset, on the right from the harder dataset, where two clusters largely overlap. The covariance in the plots above is $\sigma^2 = 0.1$.

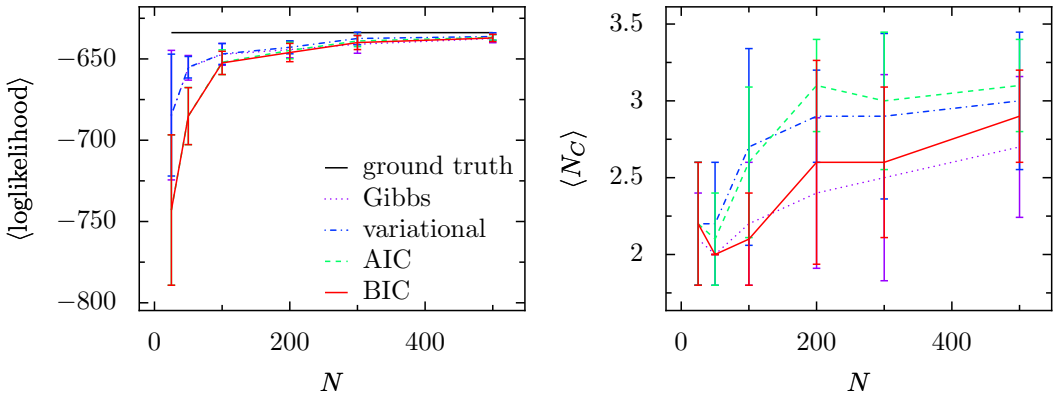
The results on the Zurich dataset are shown in Figure 4.3. We observe that the different model order selection algorithms perform around equally. However, the expectation of the log likelihood was consistently smaller for the Dirichlet process methods on small datasets, where the σ^2 parameter is not very large, but this was only within standard deviation.



(a) easier configuration, $\sigma^2 = 0.05$



(b) easier configuration, $\sigma^2 = 0.4$



(c) harder configuration, $\sigma^2 = 0.05$

Figure 4.3: The first and second row show results on the easier Zurich dataset for different values of σ^2. The third row shows a result from the harder dataset. As we can clearly see in the plot of the expected number of clusters for the harder dataset, the number of clusters approaches 3, the more data we see.

Berne dataset: 7 clusters

The Berne dataset characteristics and our experiments are summarized in Table 4.2. It's basically the same as the Zurich dataset, but here we have 7 clusters, instead of only 3.

type	Gaussian: $\boldsymbol{\mu}_\theta = [0, 0], \boldsymbol{\Sigma}_{\boldsymbol{\theta}} = 3\boldsymbol{I}, \boldsymbol{\Sigma}_{\boldsymbol{x}} = \sigma^2\boldsymbol{I}$
dimension	2
number of clusters N_C	7
balanced	yes
σ^2	$0.05, 0.1, 0.2, 0.3, 0.4$
α	0.2
N	$25, 50, 100, 200, 300, 500$
repetitions	10
algorithms run	collapsed Gibbs, variational, AIC, BIC

Table 4.2: The Berne dataset and the experiments we performed on it.

We show samples from the two different configurations of the Berne dataset in Figure 4.4.

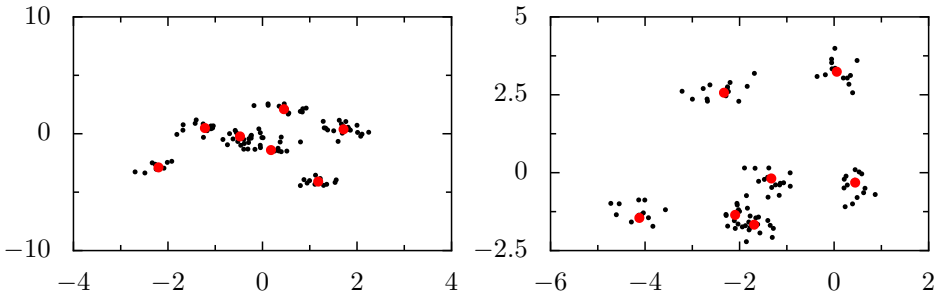
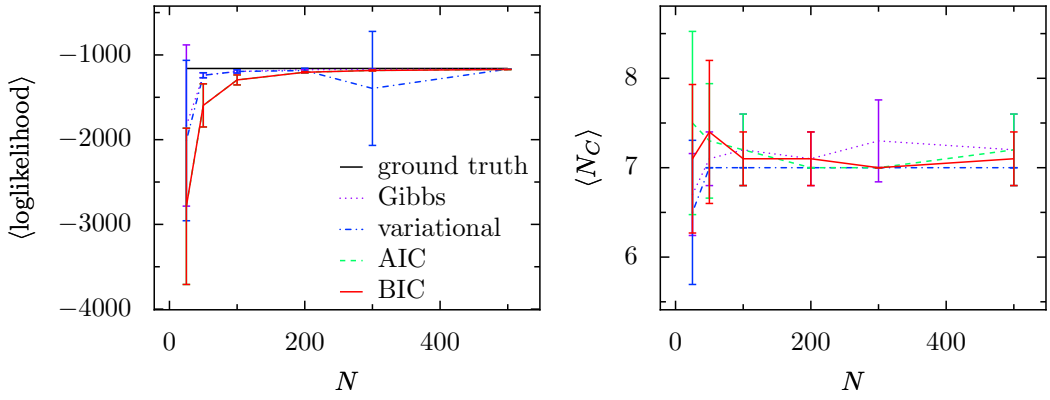
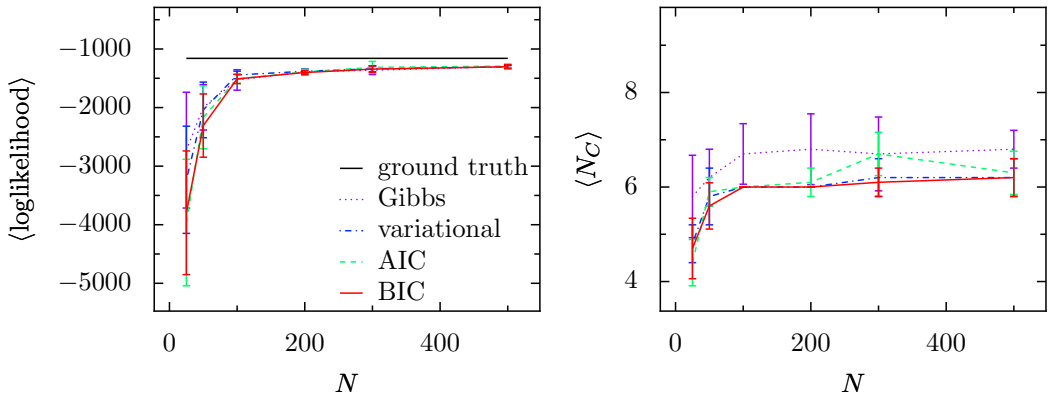


Figure 4.4: On the left we see two samples from the easier Zurich dataset, on the right from the harder dataset, where two clusters largely overlap. The covariance in the plots above is $\sigma^2 = 0.1$.

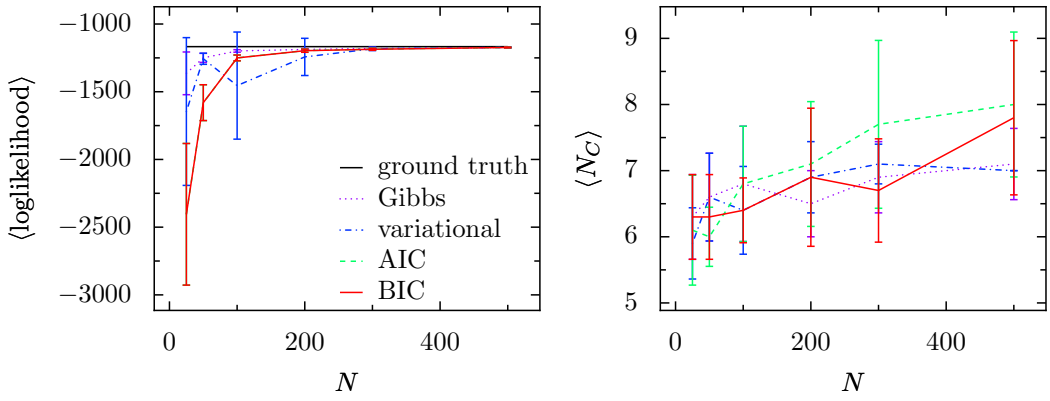
The results on the Berne dataset are shown in Figure 4.5. We observe that the different model order selection algorithms perform around equally. Again, the expectation of the log likelihood was consistently smaller for the Dirichlet process methods on small datasets, where the σ^2 parameter is not very large, but this was only within standard deviation.



(a) easier configuration, $\sigma^2 = 0.05$



(b) easier configuration, $\sigma^2 = 0.4$



(c) harder configuration, $\sigma^2 = 0.05$

Figure 4.5: The first and second row show results on the easier Berne dataset for different values of σ^2. The third row shows a result from the harder dataset.

London dataset: higher dimensionality

The London dataset characteristics and our experiments are summarized in Table 4.3. Again, it's basically the same as the Zurich dataset, but here we have a dimensionality of 8, instead of 2.

type	Gaussian: $\boldsymbol{\mu}_\theta = [0, \ldots, 0], \boldsymbol{\Sigma}_{\boldsymbol{\theta}} = 3\boldsymbol{I}, \boldsymbol{\Sigma}_{\boldsymbol{x}} = \sigma^2 \boldsymbol{I}$
dimension	8
number of clusters N_C	5
balanced	yes
σ^2	$0.1, 0.2, 0.3, 0.6, 0.8$
α	0.2
N	$100, 200, 300, 500, 700$
repetitions	10
algorithms run	collapsed Gibbs, variational, AIC, BIC

Table 4.3: The London dataset and the experiments we performed on it.

The results on the London dataset are shown in Figure 4.6. Also on this data set, all of the algorithms perform about equal.

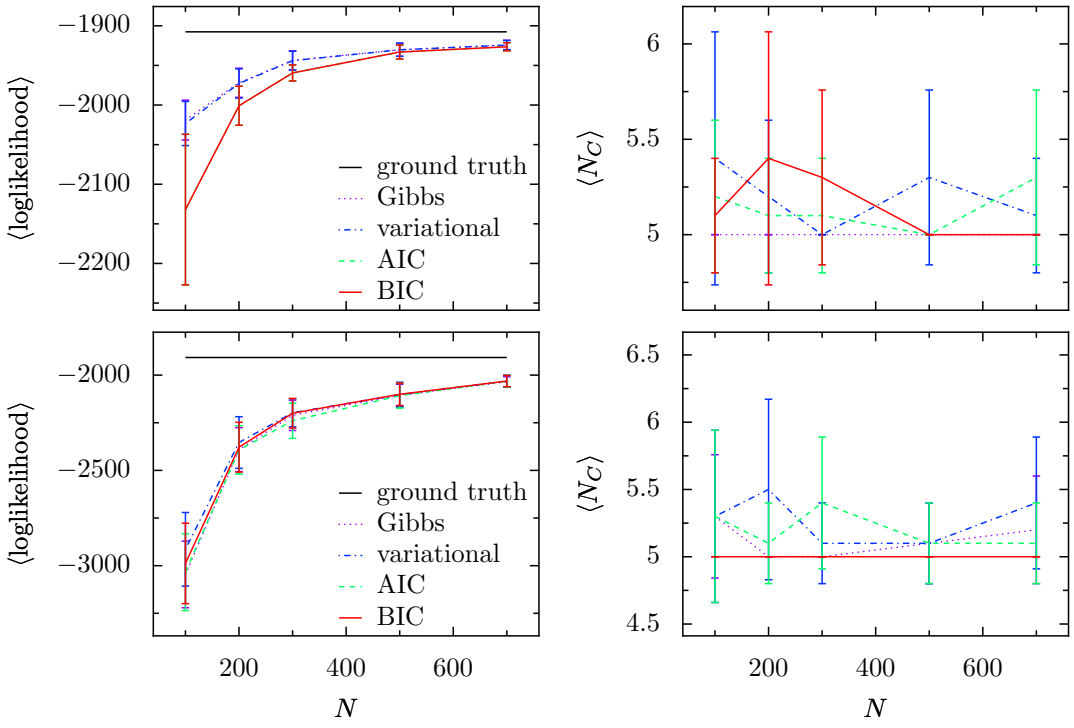


Figure 4.6: The results on the London dataset for two different values of σ^2 (top: $\sigma^2 = 0.1$, bottom: $\sigma^2 = 0.8$). Again we observe a slight advantage of the DPM on small datasets with small covariance.

Delhi dataset: unbalanced clusters

Here we changed the size of the different clusters. From looking at data generated by a Dirichlet process, one would assume that the DP performs very well on this kind of data, as it often is unbalanced. Surprinsingly, again we did not see much of a difference.

type	Gaussian: $\boldsymbol{\mu}_\theta = [0, 0], \boldsymbol{\Sigma}_{\boldsymbol{\theta}} = 3\boldsymbol{I}, \boldsymbol{\Sigma}_{\boldsymbol{x}} = \sigma^2\boldsymbol{I}$
dimension	2
number of clusters N_C	3
balanced	no: (1) [0.66, 0.11, 0.23], (2) [0.67, 0.0813, 0.2464]
σ^2	$0.05, 0.1, 0.2, 0.3, 0.4$
α	0.2
N	$25, 50, 100, 200, 300, 500$
repetitions	10
algorithms run	collapsed Gibbs, variational, AIC, BIC

Table 4.4: The Delhi dataset and the experiments we performed on it.

The results on the Delhi dataset are shown in Figure 4.7.

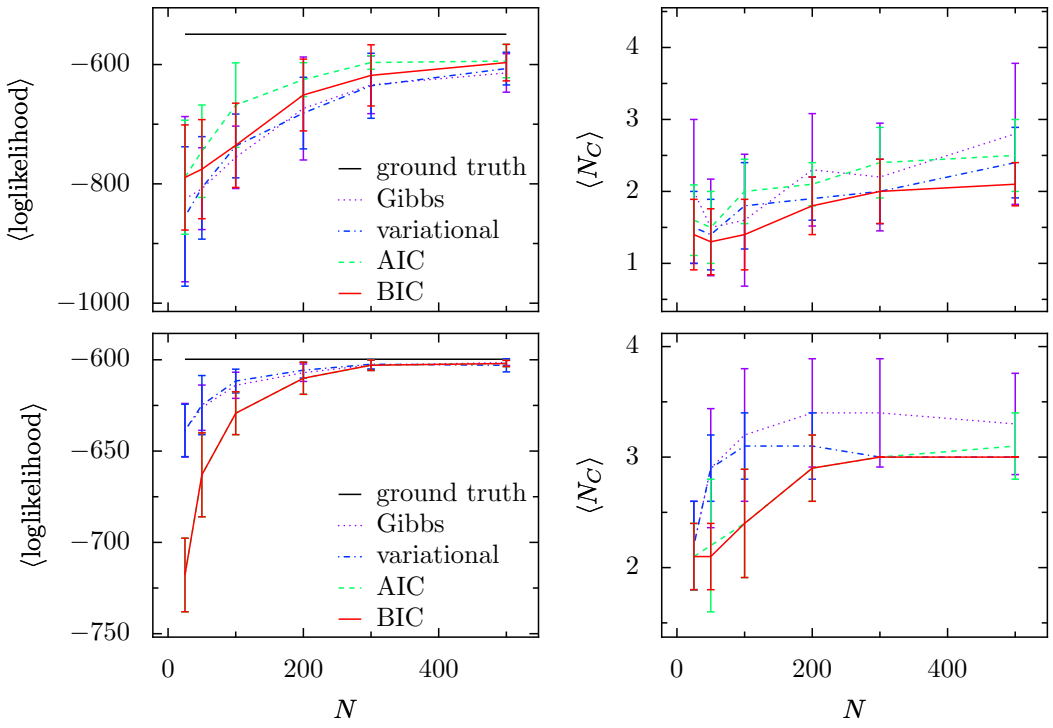


Figure 4.7: The results on the Delhi dataset. Top: easier configuration with $\sigma^2 = 0.4$. The DPM methods seem slightly inferior. Bottom: harder configuration with $\sigma^2 = 0.05$.

Rome dataset: high dimensionality and many clusters

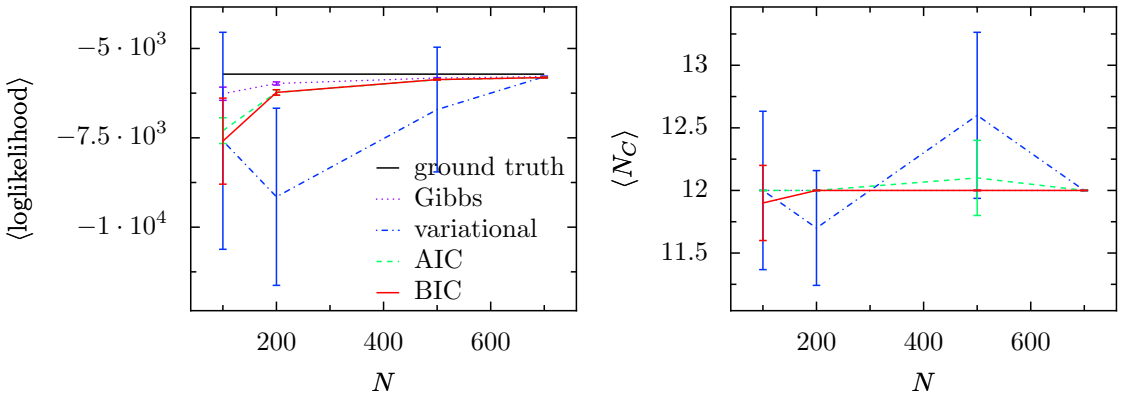
So far, there was not much of a difference between the different MOS algorithms. In this data set we further increase the number of clusters and the dimensionality.

type	Gaussian: $\boldsymbol{\mu}_\theta = [0, \ldots, 0], \boldsymbol{\Sigma}_{\boldsymbol{\theta}} = 3\boldsymbol{I}, \boldsymbol{\Sigma}_{\boldsymbol{x}} = \sigma^2 \boldsymbol{I}$
dimension	12
number of clusters N_C	15
balanced	yes
σ^2	$0.2, 0.3, 0.4$
α	1
N	$100, 200, 500, 700$
repetitions	10
algorithms run	collapsed Gibbs, variational, AIC, BIC

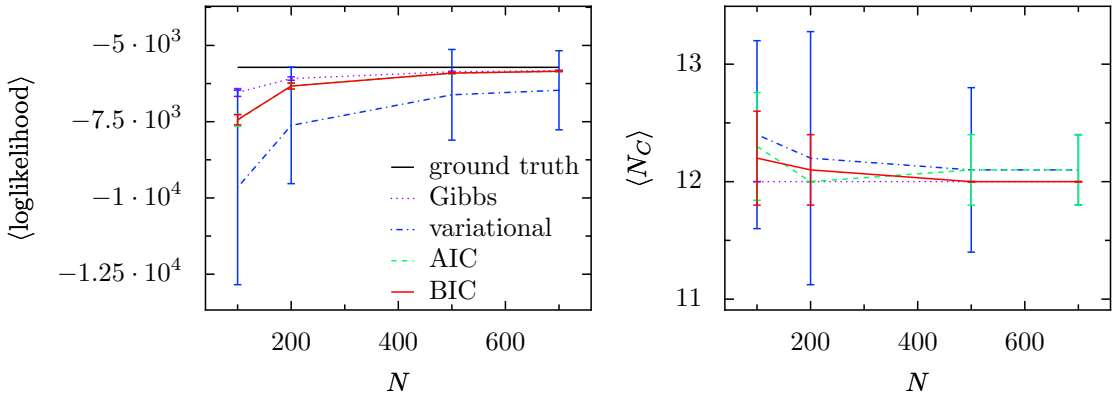
Table 4.5: The Rome dataset and the experiments we performed on it.

The results on the Rome dataset are shown in Figure 4.8.

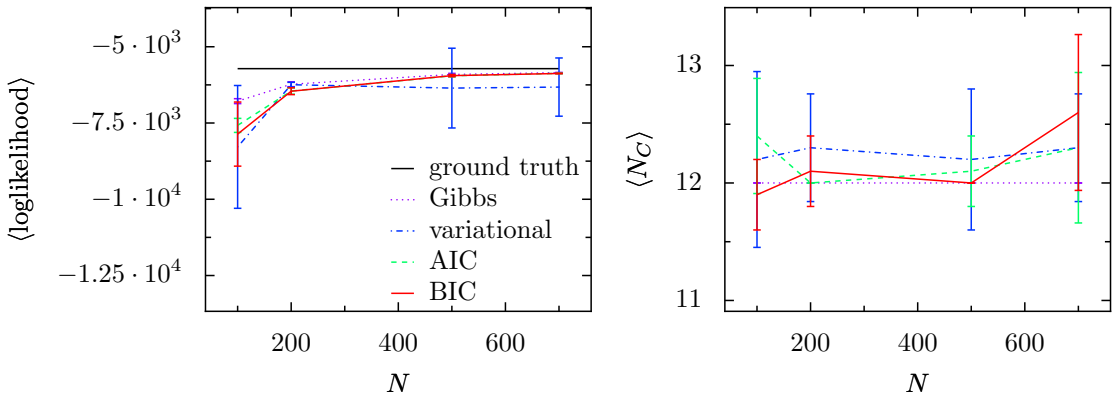
Looking at the results of the variational algorithm, we observed that for some of the repetitions (on newly sampled data) the variational algorithm led to much worse likelihoods on the validation set; it seems to converge to local minima. Investigating further we found out that neither the dimensionality nor the number of clusters itself is a problem, but rather the combination of the two seems to lead to problems. We could improve on the results shown above by increasing the number of iterations of the variational algorithm and increasing the upper bound on the number of clusters, however it still happened that some of the repetitions of the variational algorithm converged to configurations with considerably smaller likelihood than what the other algorithms returned. We also checked the variational inference implementations from [Kurihara et al., 2007b,c], which are freely available from the first author's webpage, however these algorithms show similar problems, while on simpler problems they lead to almost the same result as our variational implementation.



(a) $\sigma^2 = 0.2$



(b) $\sigma^2 = 0.3$



(c) $\sigma^2 = 0.4$

Figure 4.8: The results for the Rome dataset.

4.4.2 Dirichlet/Multinomial

So far we have considered Gaussian data, we now switch to histogram data, which is another very important model for certain applications.

Boston dataset: 3 clusters

The Boston dataset characteristics and our experiments are summarized in Table 4.6.

type	Multinomial, Dirichlet prior: $\boldsymbol{\alpha} = [1, 1, 1, 1]$
dimension	4
number of clusters N_C	3
balanced	yes
M	15
α	0.2
N	$25, 50, 100, 200, 300, 500$
repetitions	10
algorithms run	collapsed Gibbs, variational, AIC, BIC

Table 4.6: The Boston dataset and the experiments we performed on it.

We show the cluster means from the two different configurations of the Boston dataset in Figure 4.9.

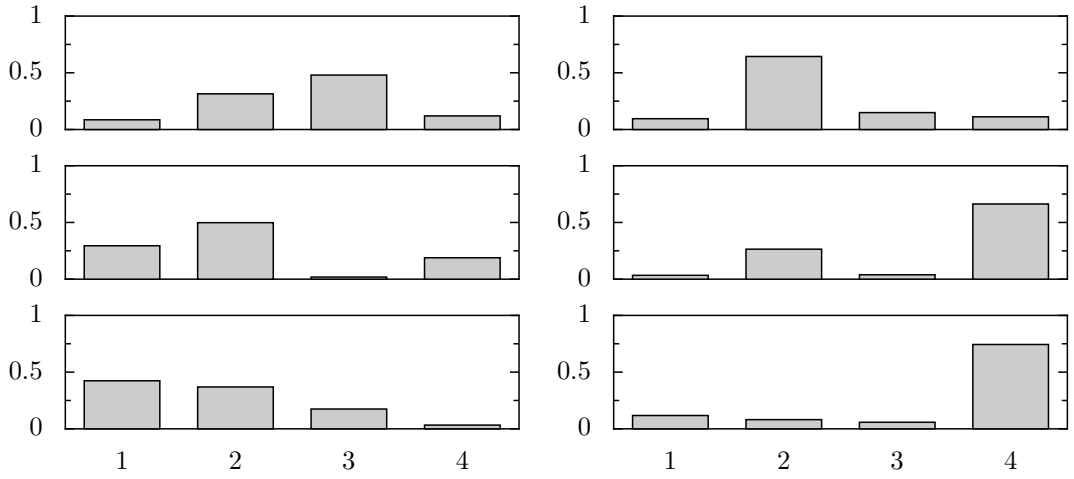
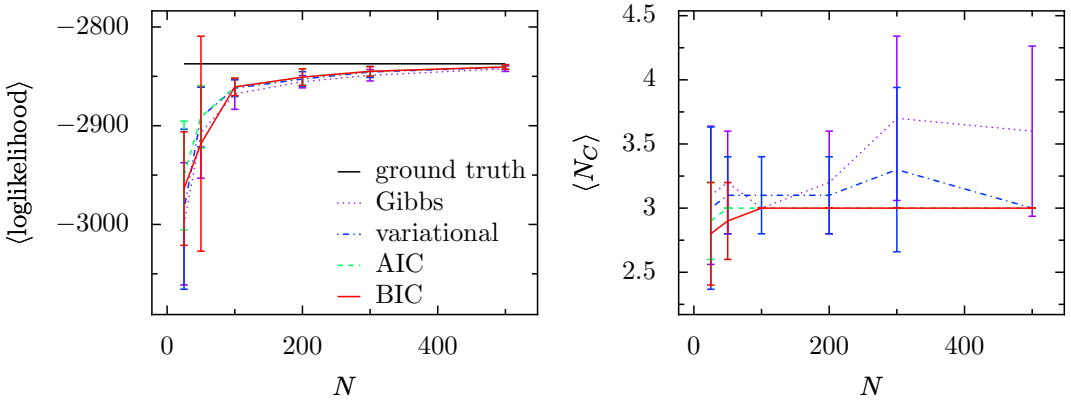
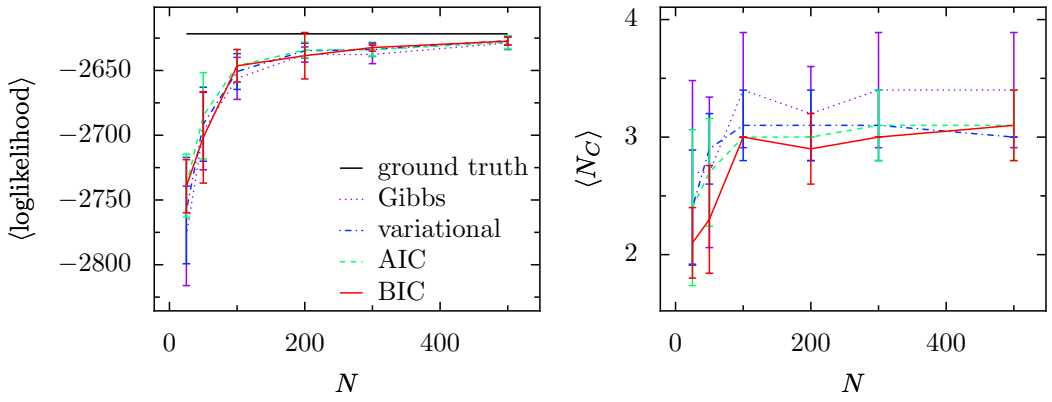


Figure 4.9: On the left we see the cluster parameters of the three clusters for the simpler configuration, on the right for the slightly harder configuration.

The results on the Boston dataset are shown in Figure 4.10. Again, as for the Gaussian data sets, we observe that the different model order selection algorithms perform around equally.



(a) easier configuration



(b) harder configuration

Figure 4.10: The first row shows the result on the easier configuration, while the second row shows the result on the slightly harder configuration.

Toronto: 7 clusters

The Toronto dataset is similar to the Boston dataset, however here we consider 7 clusters, instead of only 3. The characteristics and our experiments are summarized in Table 4.7.

type	Multinomial, Dirichlet prior: $\boldsymbol{\alpha} = [1, 1, 1, 1]$
dimension	4
number of clusters N_C	7
balanced	yes
M	15
α	0.8
N	$25, 50, 100, 200, 300, 500$
repetitions	10
algorithms run	collapsed Gibbs, variational, AIC, BIC

Table 4.7: The Toronto dataset and the experiments we performed on it.

As before, there's not much of a difference between the different algorithms for the log likelihood of the inferred model on the validation set.

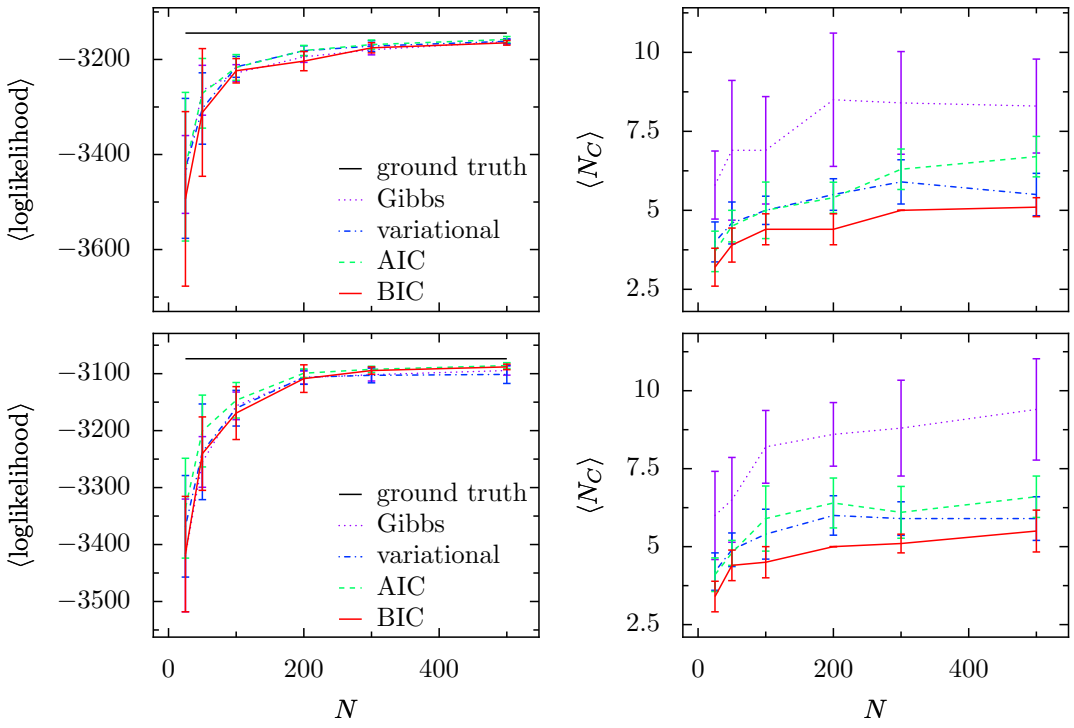


Figure 4.11: Results on the Toronto data set. Top: easier configuration, bottom: harder configuration.

New York: Higher dimensionality

In this data set we increase the dimensionality of the problem. The New York dataset characteristics and our experiments are summarized in Table 4.8.

type	Multinomial, Dirichlet prior: $\boldsymbol{\alpha} = [1, \ldots, 1]$
dimension	10
number of clusters N_C	5
balanced	yes
M	15
α	0.4
N	$25, 50, 100, 200, 300, 500$
repetitions	10
algorithms run	collapsed Gibbs, variational, AIC, BIC

Table 4.8: The New York dataset and the experiments we performed on it.

For small datasets the Gibbs algorithm showed to be inferior when compared with the other algorithms.

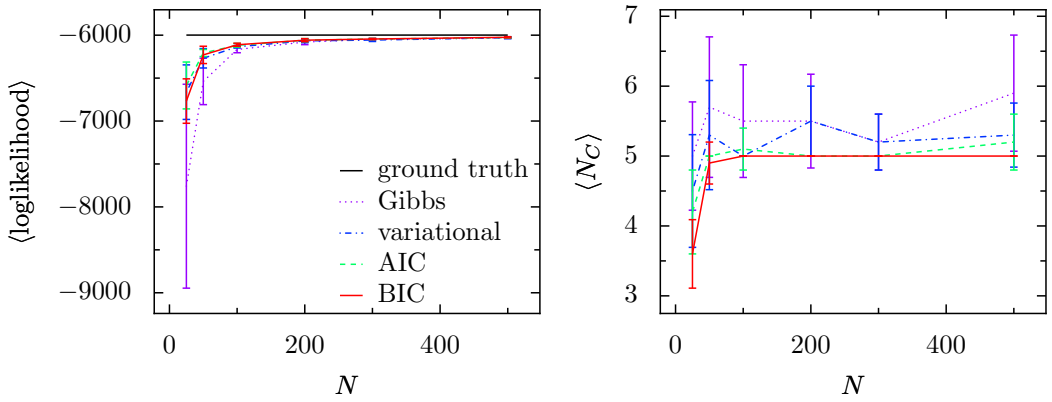


Figure 4.12: Results on the New York data set.

Los Angeles: unbalanced clusters

This is data set is unbalanced, and some clusters are more important than others. The Los Angeles dataset characteristics and our experiments are summarized in Table 4.9.

type	Multinomial, Dirichlet prior: $\boldsymbol{\alpha} = [1, 1, 1, 1]$
dimension	4
number of clusters N_C	3
balanced	no: [0.713,0.249,0.039] and [0.013,0.034,0.953]
M	15
α	0.2
N	$25, 50, 100, 200, 300, 500$
repetitions	10
algorithms run	collapsed Gibbs, variational, AIC, BIC

Table 4.9: The Los Angeles dataset and the experiments we performed on it.

For the harder configuration, the log likelihood of the BIC algorithm is slightly smaller than of the other methods.

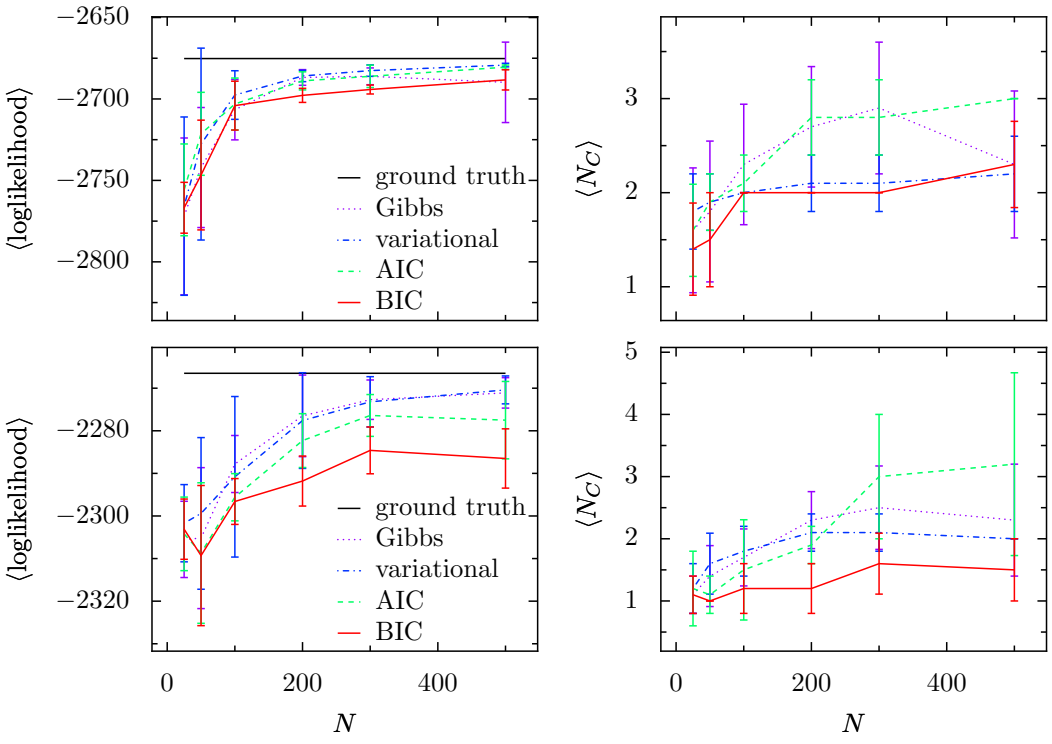


Figure 4.13: Results on the Los Angeles data set. Top: easier configuration, bottom: harder configuration.

San Francisco: higher dimensionality and many clusters

In this dataset we further increased the dimension and the number of clusters. The San Francisco dataset characteristics and our experiments are summarized in Table 4.10.

type	Multinomial, Dirichlet prior: $\boldsymbol{\alpha} = [1, \ldots, 1]$
dimension	20
number of clusters N_C	12
balanced	yes
M	15
α	2
N	$25, 50, 100, 200, 300, 500$
repetitions	10
algorithms run	collapsed Gibbs, variational, AIC, BIC

Table 4.10: The San Francisco dataset and the experiments we performed on it.

For smaller datasets the Gibbs algorithm performs quite poor. This already happened in the New York data set and as both are datasets with higher dimensionality, it looks as if the Gibbs algorithm and higher dimensional histogram is a bad combination for small datasets.

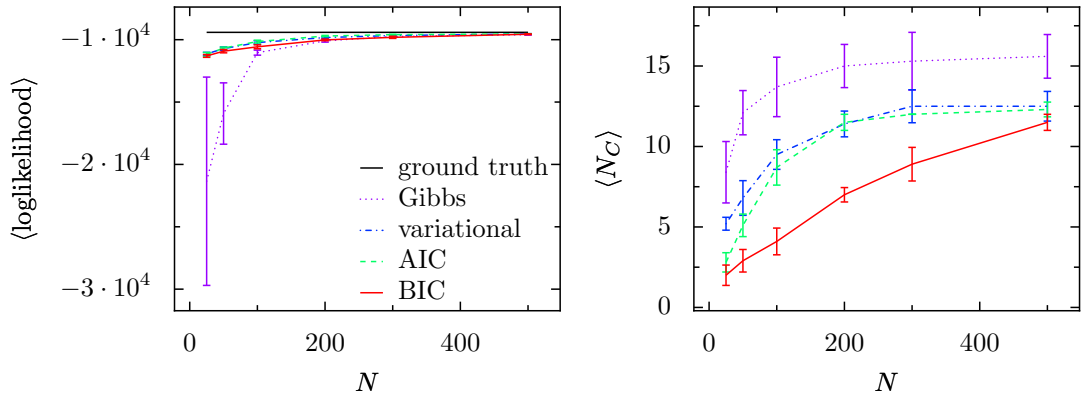


Figure 4.14: Results on the San Francisco dataset.

4.5 Conclusions

The general outcome of the comparison is that there is not a huge difference of the various MOS strategies considered on our synthetic data sets for the log likelihood. However, if one considers the number of clusters inferred there is a bigger difference. As BIC usually adds clusters in a very conservative way the number of clusters inferred is sometimes underestimated for smaller data sets. On the other hand, the BIC score is usually very steady and the variance is often smaller than for the other methods. The

variational inference for the DP, except for the Rome data set, shows to be very good, if one considers the log likelihood. The number of clusters inferred is more dynamic than with the BIC, but still considerably fewer clusters than with the Gibbs algorithm are estimated. The Gibbs algorithm usually overestimates the number of clusters and also shows a large variance, however this seems to not have a bigger impact on the log likelihood of the inferred model. On small data sets, it seems as if the DP methods would have a small advantage, as they often return a smaller log likelihood for $N \leq 100$. Initially, one fear was that the Dirichlet process methods would perform considerably worse, as nice theories sometimes perform bad in practice. At least in our experiments, we did find no evidence that the DP would be inferior to the more traditional and popular information theoretic methods.

5 Nonparametric Bayesian Biclustering

> They say the devil's water. It ain't so sweet. You don't have to drink right now. But you can dip your feet. Every once in a little while.
>
> *(The Killers in 'When You Were Young')*

So far we've discussed various models and algorithms that have been proposed to tackle the clustering problem. While the discussion of clustering methods did include data of arbitrary dimensions (N samples of dimension d), the clustering problem itself always was one-dimensional: we assign exactly one cluster to each of the N data points. In this chapter we discuss a (restricted) extension to 2D, usually referred to as *biclustering* or sometimes also *co-clustering*. Informally, the goal is to cluster both, the objects (the rows of $\boldsymbol{X}$) and the features (the columns of $\boldsymbol{X}$) simultaneously for an appropriately defined cost function.

Definition 5.1 (Biclustering, Bicluster)**.** Given a data matrix $\boldsymbol{X}$ of dimension $N \times d$. A *biclustering* of this matrix corresponds to an object partition $\mathcal{C}^{\mathcal{O}} = \{\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_{N_C^{\mathcal{O}}}\}$ and a feature partition $\mathcal{C}^{\mathcal{F}} = \{\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_{N_C^{\mathcal{F}}}\}$. We formally define the partition of the objects as follows:

$$\begin{aligned} \mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_{N_C^{\mathcal{O}}}, \quad \mathcal{O}_\mu \subseteq \{1, \ldots, N\}, \quad \mu = 1, \ldots, N_C^{\mathcal{O}}, \\ \mathcal{O}_1 \cup \mathcal{O}_2 \cup \ldots \cup \mathcal{O}_{N_C^{\mathcal{O}}} = \{1, \ldots, N\}, \\ \mathcal{O}_\mu \cap \mathcal{O}_{\mu'} = \emptyset, \quad \mu, \mu' = 1, \ldots, N_C^{\mathcal{O}}, \quad \mu \neq \mu'. \end{aligned}$$

And equivalently for the features as:

$$\begin{aligned} \mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_{N_C^{\mathcal{F}}}, \quad \mathcal{F}_\nu \subseteq \{1, \ldots, d\}, \quad \nu = 1, \ldots, N_C^{\mathcal{F}}, \\ \mathcal{F}_1 \cup \mathcal{F}_2 \cup \ldots \cup \mathcal{F}_{N_C^{\mathcal{F}}} = \{1, \ldots, d\}, \\ \mathcal{F}_\nu \cap \mathcal{F}_{\nu'} = \emptyset, \quad \nu, \nu' = 1, \ldots, N_C^{\mathcal{F}}, \quad \nu \neq \nu'. \end{aligned}$$

A *bicluster* (μ, ν) refers to the elements of $\boldsymbol{X}$ indexed by the intersection of the rows as specified by $\mathcal{O}_\mu$ and the columns as specified by $\mathcal{F}_\nu$. For given $N_C^{\mathcal{O}}$ and $N_C^{\mathcal{F}}$ there exist $N_C^{\mathcal{O}} \times N_C^{\mathcal{F}}$ biclusters.

An example of a simple biclustering is shown in Figure 5.1. As the example suggests, we are usually not interested in an arbitrary biclustering, but in one where the elements within a bicluster are similar. Also, we want to minimize the number of biclusters. This is a typical *optimization problem* as often encountered in computer science: on one

	a	b	b	a	a	a	b
a	4	1	0	5	4	5	0
b	7	10	9	6	7	7	10
a	6	0	0	4	4	4	1
c	13	4	5	12	13	12	4
a	5	1	1	4	5	5	0

	a	a	a	a	b	b	b
a	4	5	4	5	1	0	0
a	6	4	4	4	0	0	1
a	5	4	5	5	1	1	0
b	7	6	7	7	10	9	10
c	13	12	13	12	4	5	4

Figure 5.1: An example of biclustering. Left: the data matrix with the clustering of the rows and columns indicated by chars. Right: the same matrix, but reordered according the cluster assignments, we see that all the elements of the 6 biclusters have similar numbers.

end we have the biclustering where each row and column is in its own cluster and the biclusters each have size 1×1 (many biclusters, "infinite" similarity within the biclusters) and on the other end we have the biclustering that assigns all rows and all columns to one cluster (only one bicluster, but possibly very small similarity within the bicluster). So far we have not given a formal definition of what we mean with similar, we refrained from doing so because there does not exist one unique similarity measure. It needs to be specified depending on the application and the underlying model. As with many optimization problems, in general the problem is NP-hard [Garey and Johnson, 1979].

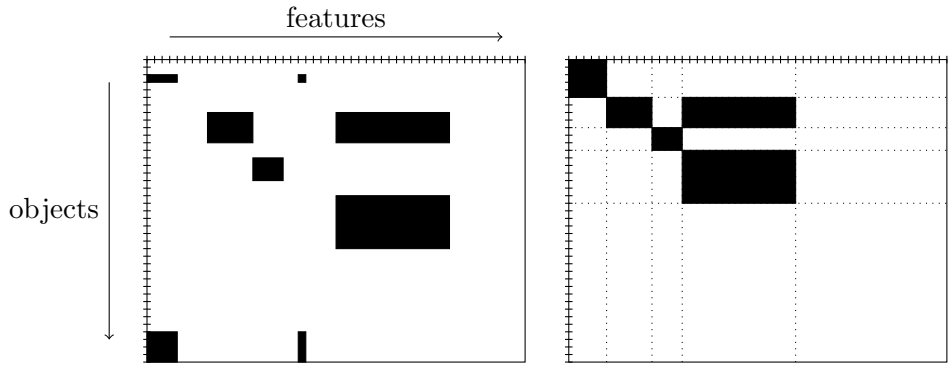


Figure 5.2: Left: sparse input data matrix where the non-zero elements are shown in black. To the right we see the same data, with the rows and columns rearranged to reveal the biclusters. Note that the input is normally not already that nicely structured, see for example Figure 5.3 for a harder configuration.

It should have become obvious, that the biclustering problem can *not* be regarded as two independent instances of the clustering problem in general, due to its 2D structure. Nevertheless, for some data models with restrictive assumptions, this reduction holds. We discuss this in section 5.1.1.

Biclustering can also be interpreted in a graph-theoretic sense. Let us define a bipartite

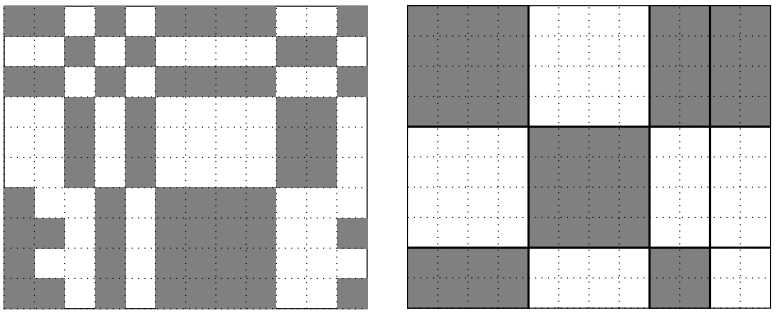


Figure 5.3: Input to the left and rearranged matrix to the right showing the biclusters.

graph $G = (O \cup F, E)$, where $O = \{1, \ldots, N\}$ (the objects), $F = \{1, \ldots, d\}$ (the features) and $E_{ij} = X_{ij}$ for $i \in \{1, \ldots, N\}$ and $j \in \{1, \ldots, d\}$. The nodes of the graph are given by $O \cup F$, the undirected edges by E. Every object i is connected to every feature j by an edge E_{ij}. The biclustering corresponds to partitioning the graph into bicliques, see Figure 5.4. The *cost* associated with a particular biclustering, can then be formulated by an appropriately defined cut function. For more background information see for example Dhillon [2001] and Busygin et al. [2005].

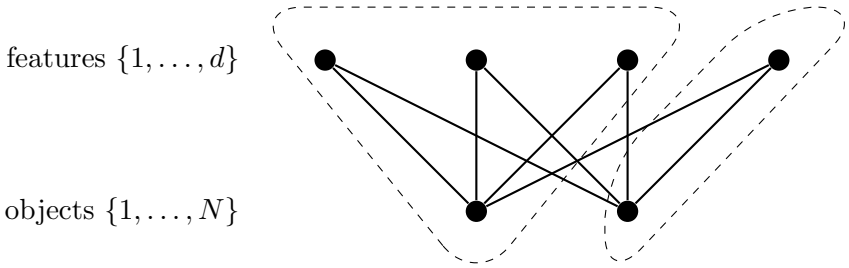


Figure 5.4: Expressing biclustering as a graph theoretic problem. Here for 2 objects and 4 features We indicate a possible partitioning of the bipartite graph into two bicliques. This corresponds to 4 biclusters, two of them of size 1×3, the others have size 1×1.

The most important application of biclustering these days is in Biology [Cheng and Church, 2000, Madeira and Oliveira, 2004, Prelić et al., 2006], where it is prominently used for clustering micro-array data (the genes correspond to the objects, conditions to the features). However, there exist also applications in other areas, for example in information retrieval and natural language processing [Dhillon, 2001] (which words are important for a document corpus?) or in computer vision (which colors are dominant for the background?).

Although, unlike clustering, biclustering does not seem very popular in the machine learning community, several methods (parametric and nonparametric) have been pro-

posed for this purpose. Bock [2003] assumes the number of clusters for the objects and features is known and maximizes the ψ-divergence of Csiszár between the co-occurrence of a bicluster and the individual, independent clusters; the Kullback-Leiber divergence is for example an instance of the ψ-divergence. More recently Seldin et al. [2007] proposed a method based on the information bottleneck: the objective is very similar to the one of Bock. The approach of Seldin et al. does however not assume that the number of clusters is known; they use a minimum description length (MDL) strategy for inferring the number of object and feature clusters. Approaches based on the Dirichlet process [Kemp et al., 2006, Kurihara et al., 2006, 2007a] have also been proposed recently, and will be discussed later on. Another related approach is given in [Meeds et al., 2007], the authors do not consider biclustering, but latent feature analysis, and as a nonparametric prior they use the Indian buffet process. In their approach it is possible for the clusters to overlap, which is sometimes desirable for applications in biology.

Depending on the model assumptions for biclustering, unsupervised feature selection [Law et al., 2003, Roth and Lange, 2004] shares similar goals, such as: How important is a feature j for object class μ? Or how similar are two features j and j'? However, there's a key difference between the two: While, in the context of biclustering, we are interested in grouping or clustering features together, in feature selection, as the name implies, we also want to select the discriminative features. However, assuming the quality of the inferred feature grouping is good, one can readily derive a criterion for the selection of the features: we should select only one feature per feature class, or could even build an average of all the features within a class and thus construct a new feature. It's important to realize the asymmetric nature of the feature clustering problem: we have objects which show (depending on the cluster to which they belong to) certain properties, called features, and as it so happens, we represent these objects in a matrix. In contrast, a general biclustering model should arguably be defined on the whole matrix, which does not allow such an asymmetric interpretation anymore.

Despite being more complex than standard clustering, most of the concepts from clustering apply directly to biclustering. In particular also the problem of *model order selection*, which we have discussed extensively throughout this thesis. In this chapter we are interested in the application of MOS ideas from standard clustering to biclustering. To that end, we first introduce a nonparametric mixture model for biclustering and discuss various specialized instances of this model. In the second part we also derive Gibbs sampling algorithms for these models.

5.1 Statistical models for biclustering

In our discussion we will restrict ourselves to instances, where the elements of the data matrix $\boldsymbol{X}$ are *discrete*. If we would encounter continuous values in practice, we could usually fulfill this assumption by rescaling and rounding our data. Furthermore, we focus on describing the different models and assumptions from a statistical and machine

learning point of view, this might differ from literature in other communities. Most of the approaches described here, show similarities and are in fact inspired by models that were already proposed for clustering *co-occurrence data* [Hofmann and Puzicha, 1998]. In the co-occurrence setting, one however assumes less structure in that an object can be assigned to more than one object cluster or a feature can be assigned to more than one feature cluster. In our discussion we will restrict ourselves to biclustering models with *uniform distribution* within a bicluster. Most biclustering algorithms have this assumption in one form or the other; it formalizes the assumption of a coherent "block".

5.1.1 Symmetric models for biclustering

In this subsection we introduce two symmetric, generative models for biclustering. The nomenclature of distinguishing objects and features is unfortunate for this part, as it already implies an asymmetry in the problem, which is not desired here.

Let's denote the total number of counts in the matrix $\boldsymbol{X}$ by $R = \sum_{i,j} X_{ij}$ and assume that the matrix has dimension $N \times d$. For the symmetric models to come, it is easier to represent our data as a serialized version of $\boldsymbol{X}$; we use the notation of Hofmann and Puzicha: the atomic entities we'll be studying are two finite sets $\mathcal{X} = \{x_1, \ldots, x_N\}$ (the set of objects) and $\mathcal{Y} = \{y_1, \ldots, y_d\}$ (the set of features). As elementary observations we consider pairs $(x_i, y_j) \in \mathcal{X} \times \mathcal{Y}$, i.e. a joint occurrence of object x_i and feature y_j. All data is numbered and collected in a sample set $\mathcal{S} = \{(x_{i(r)}, y_{j(r)}, r) \ : \ 1 \leq r \leq R\}$ with arbitrary ordering.

Infinite symmetric biclustering model

We can explain the observed data $\mathcal{S}$ by the infinite symmetric biclustering model (ISBM) below.

$$\begin{aligned}
\boldsymbol{z}^{\mathcal{O}} \,|\, \alpha^{\mathcal{O}} &\sim \mathrm{CRP}(\alpha^{\mathcal{O}}) \\
\boldsymbol{z}^{\mathcal{F}} \,|\, \alpha^{\mathcal{F}} &\sim \mathrm{CRP}(\alpha^{\mathcal{F}}) \\
\boldsymbol{\Pi} \,|\, \beta, \boldsymbol{z}^{\mathcal{O}}, \boldsymbol{z}^{\mathcal{F}} &\sim \mathrm{Dir}(\boldsymbol{\Pi}|\boldsymbol{\Lambda}) \\
(\mu_r, \nu_r) \,|\, \boldsymbol{\Pi} &\sim \mathrm{Mult}((\mu_r, \nu_r)|\boldsymbol{\Pi}, 1) \\
x_{i(r)} \,|\, \boldsymbol{z}^{\mathcal{O}}, \mu_r &\sim \mathrm{Mult}(x_{i(r)}|\boldsymbol{\theta}^{\mathcal{O}_{\mu_r}}, 1) \\
y_{j(r)} \,|\, \boldsymbol{z}^{\mathcal{F}}, \nu_r &\sim \mathrm{Mult}(y_{j(r)}|\boldsymbol{\theta}^{\mathcal{F}_{\nu_r}}, 1).
\end{aligned} \tag{5.1}$$

Here $\boldsymbol{\Lambda}$ is a matrix of dimension $N_C^{\mathcal{O}} \times N_C^{\mathcal{F}}$ which is (deterministically) constructed as follows:

$$\Lambda_{\mu,\nu} = \beta m_\mu^{\mathcal{O}} m_\nu^{\mathcal{F}},$$

where we assume a uniform prior distribution over the bins (i, j) and β expresses our prior belief (per bin). $m_\mu^{\mathcal{O}}$ and $m_\nu^{\mathcal{F}}$ denote the number of objects and features assigned to object cluster μ and feature cluster ν, respectively. Furthermore each of the $\boldsymbol{\theta}^{\mathcal{O}_\mu}$ and $\boldsymbol{\theta}^{\mathcal{F}_\nu}$ for $\mu = 1, \ldots, N_C^{\mathcal{O}}$ and $\nu = 1, \ldots, N_C^{\mathcal{F}}$ is a probability vector of dimension N or

d, specifying the probability of object i or feature j for an object cluster μ or feature cluster ν, respectively. These probability vectors are defined as follows:

$$\theta_i^{\mathcal{O}_\mu} = \begin{cases} \frac{1}{m_\mu^{\mathcal{O}}} & \text{if } z_i^{\mathcal{O}} = \mu \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\theta_j^{\mathcal{F}_\nu} = \begin{cases} \frac{1}{m_\nu^{\mathcal{F}}} & \text{if } z_j^{\mathcal{F}} = \nu \\ 0 & \text{otherwise.} \end{cases}$$

The parameters $\boldsymbol{\theta}^{\mathcal{O}_\mu}$ and $\boldsymbol{\theta}^{\mathcal{F}_\nu}$ are deterministically determined by the two clusterings $\boldsymbol{z}^{\mathcal{O}}$ and $\boldsymbol{z}^{\mathcal{F}}$. This generative process can be illustrated by the graphical model in Figure 5.5.

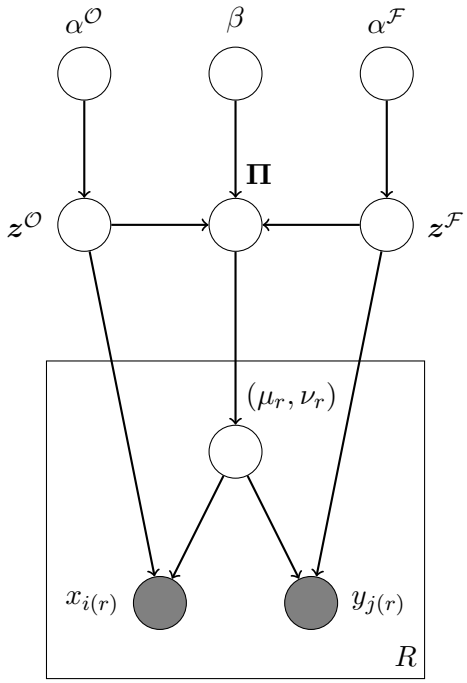


Figure 5.5: The infinite symmetric biclustering model.

In this model we first generate a clustering of the objects and features by a Chinese restaurant process, this then determines the structure (given by $N_C^{\mathcal{O}} \times N_C^{\mathcal{F}}$) and the prior belief (given by $\boldsymbol{\Lambda}$) of the probability matrix $\boldsymbol{\Pi}$, which is sampled from a Dirichlet distribution given the prior $\boldsymbol{\Lambda}$. The entry (μ, ν) of $\boldsymbol{\Pi}$ determines the probability of the bicluster (μ, ν). For each count $r = 1, \ldots, R$ in the data matrix $\boldsymbol{X}$, we then first draw a bicluster (μ_r, ν_r) and generate an object/feature pair $(x_{i(r)}, y_{j(r)})$ within the bicluster (μ_r, ν_r).

Factorized infinite symmetric biclustering model

We can simplify the biclustering model in (5.1) by assuming that $\mathbf{\Pi}$ factorizes: $\mathbf{\Pi} = \boldsymbol{\pi}^{\mathcal{O}} \times \boldsymbol{\pi}^{\mathcal{F}}$, i.e. the probability of a bicluster (μ, ν) is given by: $\Pi_{\mu,\nu} = \pi^{\mathcal{O}}_{\mu} \pi^{\mathcal{F}}_{\nu}$. $\boldsymbol{\pi}^{\mathcal{O}}$ and $\boldsymbol{\pi}^{\mathcal{F}}$ are probability vectors of size $N_C^{\mathcal{O}}$ and $N_C^{\mathcal{F}}$, respectively. Like this, the two clusterings become fully independent and can be handled separately. This is a real restriction and might show to be a serious limitation, depending on the application, as it makes it for example impossible to express the fact, that we would expect biclusters to lie along the diagonal of the matrix (whether this is a sensible assumption, largely depends on the problem at hands). The model then looks as follows:

$$\begin{aligned}
\boldsymbol{z}^{\mathcal{O}} \,|\, \alpha^{\mathcal{O}} &\sim \mathrm{CRP}(\alpha^{\mathcal{O}}) \\
\boldsymbol{z}^{\mathcal{F}} \,|\, \alpha^{\mathcal{F}} &\sim \mathrm{CRP}(\alpha^{\mathcal{F}}) \\
\boldsymbol{\pi}^{\mathcal{O}} \,|\, \beta, \boldsymbol{z}^{\mathcal{O}} &\sim \mathrm{Dir}(\boldsymbol{\pi}^{\mathcal{O}} | \boldsymbol{\lambda}^{\mathcal{O}}) \\
\boldsymbol{\pi}^{\mathcal{F}} \,|\, \beta, \boldsymbol{z}^{\mathcal{F}} &\sim \mathrm{Dir}(\boldsymbol{\pi}^{\mathcal{F}} | \boldsymbol{\lambda}^{\mathcal{F}}) \\
\mu_r \,|\, \boldsymbol{\pi}^{\mathcal{O}} &\sim \mathrm{Mult}(\mu_r | \boldsymbol{\pi}^{\mathcal{O}}, 1) \\
\nu_r \,|\, \boldsymbol{\pi}^{\mathcal{F}} &\sim \mathrm{Mult}(\nu_r | \boldsymbol{\pi}^{\mathcal{F}}, 1) \\
x_{i(r)} \,|\, \boldsymbol{z}^{\mathcal{O}}, \mu_r &\sim \mathrm{Mult}(x_{i(r)} | \boldsymbol{\theta}^{\mathcal{O}_{\mu_r}}, 1) \\
y_{j(r)} \,|\, \boldsymbol{z}^{\mathcal{F}}, \nu_r &\sim \mathrm{Mult}(y_{j(r)} | \boldsymbol{\theta}^{\mathcal{F}_{\nu_r}}, 1).
\end{aligned} \tag{5.2}$$

We illustrate this model in Figure 5.6. The variables $\boldsymbol{\lambda}^{\mathcal{O}}$ and $\boldsymbol{\lambda}^{\mathcal{F}}$ are similarly defined as in the ISBM:

$$\lambda^{\mathcal{O}}_{\mu} = \beta m^{\mathcal{O}}_{\mu} \qquad \text{and} \qquad \lambda^{\mathcal{F}}_{\nu} = \beta m^{\mathcal{F}}_{\nu}.$$

To understand the limitations of this model better, let's consider the following toy problem with 4 biclusters. Assume we are considering a matrix of dimensions $N \times d$ (with a total count R; N and d a multiple of 2), with two object clusters of size $N/2$ each and two feature clusters of size $d/2$ each. The joint distribution of the different biclusters is given in Table 5.1.

	$\mathcal{F}_1$	$\mathcal{F}_2$
$\mathcal{O}_1$	0.5	0
$\mathcal{O}_2$	0	0.5

Table 5.1: Joint distribution of the biclusters for the toy example.

When R is large, we would expect each object to occur about R/N times and each feature about R/d times. Both, the object and feature clustering will then result in one cluster each, as there is apparently no difference for the occurrence of the x_i and y_j, separately, although there is a huge difference in the co-occurrence of (x_i, y_j). This thus leads to a biclutering spanning over the whole matrix which is clearly not what we would expect. This toy example nicely demonstrates the problems associated with assuming a factorized distribution for the clusters when this is not justified by the data.

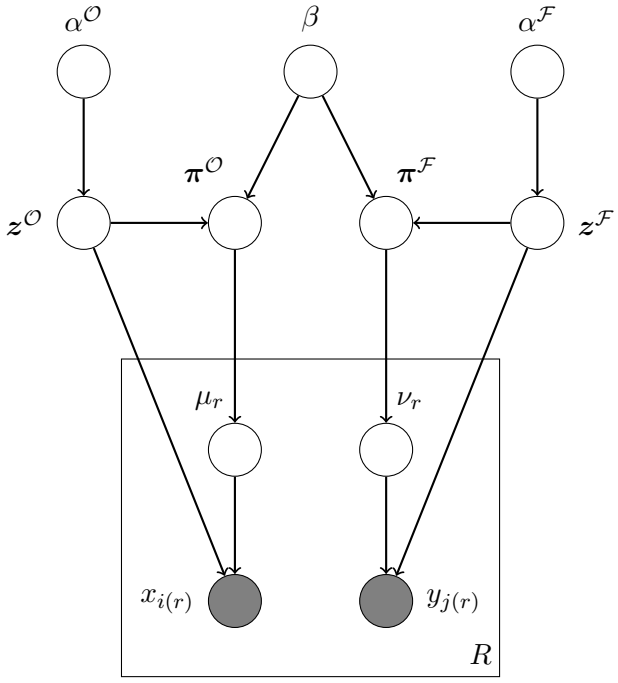


Figure 5.6: The factorized infinite symmetric biclustering model.

We won't consider this model any further, due to its limitations and its reduction to two instances of standard clustering.

5.1.2 Infinite asymmetric biclustering model

So far we've assumed that the biclustering problem is inherently symmetric: even though the functional forms of the probabilities involved, might differ, we sample for each draw a new object and feature. While this is arguably, the right approach for general biclustering, let's consider an important special case where prior knowledge about the data is available. Let's go back to the aforementioned problem of biclustering images: here, another model might be more appropriate, as the number of draws per object is constant (as these are the statistics of an image patch of fixed size, say $9 \times 9 = 81$ pixels) and we have a Multinomial model for the features (the colors) of an object. We could then instead consider the infinite asymmetric biclustering model (IABM):

1. $\boldsymbol{z}^{\mathcal{O}} \,|\, \alpha^{\mathcal{O}} \sim \mathrm{CRP}(\alpha^{\mathcal{O}})$
2. $\boldsymbol{z}^{\mathcal{F}} \,|\, \alpha^{\mathcal{F}} \sim \mathrm{CRP}(\alpha^{\mathcal{F}})$
3. $\boldsymbol{\pi}_{\mu}^{\mathcal{F}} \,|\, \beta, \boldsymbol{z}^{\mathcal{F}} \sim \mathrm{Dir}(\boldsymbol{\pi}_{\mu}^{\mathcal{F}} | \boldsymbol{\lambda}^{\mathcal{F}}) \quad \text{for } \mu = 1, \ldots, N_C^{\mathcal{O}}$
4. For each of the $z_n^{\mathcal{O}}$, repeat M times:
   a) $\nu_r \,|\, \boldsymbol{\pi}_{z_n^{\mathcal{O}}}^{\mathcal{F}} \sim \mathrm{Mult}(\nu_r | \boldsymbol{\pi}_{z_n^{\mathcal{O}}}^{\mathcal{F}}, 1)$

b) $x_{i(r)} \mid n \sim \delta_n(i)$

c) $y_{j(r)} \mid \boldsymbol{z}^{\mathcal{F}_{\nu_r}} \sim \text{Mult}(y_{j(r)} | \boldsymbol{\theta}^{\mathcal{F}_{\nu_r}}, 1)$.

We introduced a mixture vector $\boldsymbol{\pi}_\mu^{\mathcal{F}}$ for each object cluster μ. Here M is the number of draws per object, which is assumed to be constant and r is assumed to be increased by one for each newly sampled observation. Note that for the object clustering we assume a standard (infinite) mixture model. The model is *asymmetric*, as we think of the data generation as sequentially creating objects for $n = 1, \ldots, N$, assigning them to an object cluster, and based on this decision sample the feature clusters. The asymmetric model can be illustrated by means of a graphical model, as shown in Figure 5.7.

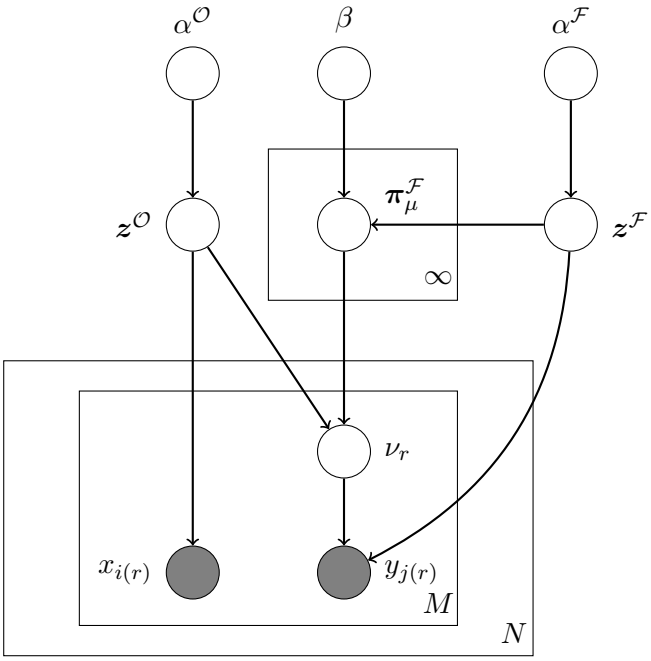


Figure 5.7: The asymmetric biclustering model. We've chosen to illustrate the generation of the $\boldsymbol{\pi}_\mu^{\mathcal{F}}$ for $\mu = 1, \ldots, N_C^{\mathcal{O}}$ by a plate with ∞ replications, to not introduce a dependence on the object clustering; this is similar to what we did in the stick breaking representation.

Under the assumption that each object has the same number of draws, this model is very similar to the ISBM. The model is inspired by the asymmetric clustering model (ACM) [Puzicha et al., 1999]. Arguably the model introduced here is, due to its asymmetric formulation, more like a feature selection approach than a general biclustering algorithm.

5.2 Existing nonparametric biclustering models

The models above introduced nonparametric approaches for biclustering. While the models gave important insights, they are not yet very useful for an actual algorithm, due to their complexity, we will tackle this problem later on. Let us however, first compare the models to two nonparametric methods, that have already been proposed in the literature. The general idea of using two Chinese restaurant processes for the object and feature clustering was first proposed in [Kemp et al., 2006]. The authors use the following generative model, which they call the infinite relational model (IRM):

$$\begin{aligned} \boldsymbol{z}^{\mathcal{O}} \,|\, \alpha^{\mathcal{O}} &\sim \mathrm{CRP}(\alpha^{\mathcal{O}}) \\ \boldsymbol{z}^{\mathcal{F}} \,|\, \alpha^{\mathcal{F}} &\sim \mathrm{CRP}(\alpha^{\mathcal{F}}) \\ \Theta_{\mu,\nu} \,|\, \beta &\sim \mathrm{Beta}(\Theta_{\mu,\nu}|\beta,\beta) \\ X_{i,j} \,|\, \boldsymbol{\Theta}, \boldsymbol{z}^{\mathcal{F}}, \boldsymbol{z}^{\mathcal{O}} &\sim \mathrm{Bernoulli}(\Theta_{z_i^{\mathcal{O}}, z_j^{\mathcal{F}}}). \end{aligned}$$

In the notation of the previous section this is a symmetric approach. It is however restricted to binary relationships, i.e. does a Siamese cat eat meat? The IRM is similar to the ISBM; however as it only considers binary relations, we don't have to fulfill normalization constraints of the parameter $\boldsymbol{\Theta}$ (the comparable parameter was called $\boldsymbol{\Pi}$ in the ISBM), which simplifies things a lot but restricts the model to binary data.

The problem of generalizing the IRM to arbitrary count data was considered by Kurihara et al. [2006, 2007a]; they propose a new model, which takes into account the frequency of a relation and which they thus call the frequency-based infinite relational model (FIRM). The model is given by the following generative process:

$$\begin{aligned} \boldsymbol{z}^{\mathcal{O}} \,|\, \alpha^{\mathcal{O}} &\sim \mathrm{CRP}(\alpha^{\mathcal{O}}) \\ \boldsymbol{z}^{\mathcal{F}} \,|\, \alpha^{\mathcal{F}} &\sim \mathrm{CRP}(\alpha^{\mathcal{F}}) \\ \Theta_{\mu,\nu} \,|\, \beta &\sim \mathrm{Beta}(\Theta_{\mu,\nu}|\beta,\beta) \\ x_{i(r)} \,|\, \boldsymbol{u}^{\mathcal{O}} &\sim \mathrm{Mult}(x_{i(r)}|\boldsymbol{u}^{\mathcal{O}}, 1) \\ y_{j(r)} \,|\, \boldsymbol{u}^{\mathcal{F}} &\sim \mathrm{Mult}(y_{j(r)}|\boldsymbol{u}^{\mathcal{F}}, 1) \\ (x_{i(r)}, y_{j(r)}) \,|\, \boldsymbol{\Theta}, \boldsymbol{z}^{\mathcal{F}}, \boldsymbol{z}^{\mathcal{O}} &\sim \mathrm{Bernoulli}(1|\Theta_{z_i^{\mathcal{O}}, z_j^{\mathcal{F}}}). \end{aligned}$$

Here $\boldsymbol{u}^{\mathcal{O}}$ and $\boldsymbol{u}^{\mathcal{F}}$ denote the probabilities of drawing an object and a feature, respectively. Note that in this model it is possible to draw negative counts, i.e. to decide in the last step to not draw a co-occurrence $(x_{i(r)}, y_{j(r)})$ and thus the r has not the same meaning as in the models discussed above, as we likely need to draw more counts than we have observations in our matrix. The authors also give a variational algorithm for the inference of this model. In our opinion this model is still restricted, as arguably we're ultimately interested in modelling a non-factorized Multinomial distribution of the biclusters, which is here approximated by a factorized Multinomial distribution of the objects and features, made approximately non-factorized by the Bernoulli draw in the

last step. It should be pointed out, that the IRM and FIRM were proposed for relational data and thus the goals might differ from ours.

We now return to the biclustering models of the previous section with a special focus towards an inference algorithm. We will first introduce an asymmetric biclustering model that could potentially be useful for unsupervised feature selection and second we introduce a symmetric biclustering model that is similar to the IRM and FIRM.

5.3 The infinite asymmetric biclustering model and the Godzilla process

We assume here that we are dealing with histogram data, i.e. multinomially distributed data and also assume that the counts per object are equal for all objects. In practice we can achieve this by rescaling the features of the objects, which will possibly destroy a lot of structure in the data.

Let's assume we are given a cluster factor $\boldsymbol{\theta}$, i.e. a probability vector, and a feature clustering vector $\boldsymbol{z}^{\mathcal{F}}$, both of these vectors have the same dimension. To generate a histogram $\boldsymbol{x}$ we would draw M times one out of the $\dim(\boldsymbol{\theta})$ bins, where bin j is drawn with probability θ_j:

$$P(\boldsymbol{x}|\boldsymbol{\theta}, M) = \frac{M!}{x_1! \cdots x_d!} \prod_{j=1}^{d} \theta_j^{x_j}. \tag{5.3}$$

As an experiment we can augment this generative process with a feature clustering $\boldsymbol{z}^{\mathcal{F}}$; in the end we will get the same probability distribution, however this illustrates our approach nicely. Let's first introduce $\bar{\boldsymbol{\theta}}$:

$$\bar{\theta}_\nu = \sum_{j \in \mathcal{F}_\nu}^{d} \theta_j \quad \text{for } \nu = 1, \ldots, N_C^{\mathcal{F}}.$$

In other words we build the sum of the elements of $\boldsymbol{\theta}$ that are assigned to the same feature cluster; we do the very same thing with $\boldsymbol{x}$ to get $\bar{\boldsymbol{x}}$. Like this we can now rewrite equation (5.3) as a two-stage process:

$$P(\boldsymbol{x}|\boldsymbol{\theta}, \boldsymbol{z}^{\mathcal{F}}, M) = \left(\frac{M!}{\bar{x}_1! \cdots \bar{x}_{N_C^{\mathcal{F}}}!} \prod_{\nu=1}^{N_C^{\mathcal{F}}} \bar{\theta}_\nu^{\bar{x}_\nu} \right) \left(\prod_{\nu=1}^{N_C^{\mathcal{F}}} \frac{\bar{x}_\nu!}{\prod_{j \in \mathcal{F}_\nu} x_j!} \prod_{j \in \mathcal{F}_\nu} \left(\frac{\theta_j}{\bar{\theta}_\nu} \right)^{x_j} \right). \tag{5.4}$$

Here $\mathcal{F}_\nu$ is the set of features, assigned to feature cluster ν (this is determined by $\boldsymbol{z}^{\mathcal{F}}$). This generative two-stage process is valid for every partition $\boldsymbol{z}^{\mathcal{F}}$ of the features. As can be readily checked, the two distributions are the same. While this might not come as much of a surprise, this insight is important for two reasons: First, it allows us to couple the generation of the data with the feature clustering and second, it also gives us a way to

make certain feature clusterings more desirable than others: we could replace the second term in equation (5.4) by say an uniform distribution among all the features assigned to the same cluster. Uneducated feature clusterings would then have small probabilities. We formalize this idea below.

What every sophisticated generative biclustering model needs to introduce, is a method for combining the two clusterings. We propose to not directly use a mixture model for the feature clustering, but rather to perturb the cluster components according to this clustering. The perturbation essentially maps the cluster factors $\boldsymbol{\theta}_\mu$, of dimensionality d, to a low dimensional space, of which the complexity is given by the number of feature clusters $N_C^{\mathcal{F}}$. Thereafter we immediately map the result back to the high dimensional space to get $\widetilde{\boldsymbol{\theta}}_\mu$. This can be seen as a lossy compression/decompression and is related to the information bottleneck as introduced in [Tishby et al., 1999].

Formally, the elements of the decryption $\widetilde{\boldsymbol{\theta}}$ are given by

$$\widetilde{\theta}_j = \bar{\theta}_{z_j^{\mathcal{F}}} / m^{\mathcal{F}}_{z_j^{\mathcal{F}}} \quad \text{for } j = 1, \ldots, d.$$

Here $m_\nu^{\mathcal{F}}$ denotes the number of features assigned to feature cluster ν. We will refer to this process as the *Godzilla process* (as it essentially flattens the "skyline" of the distribution). The steps described above are summarized in Figure 5.8.

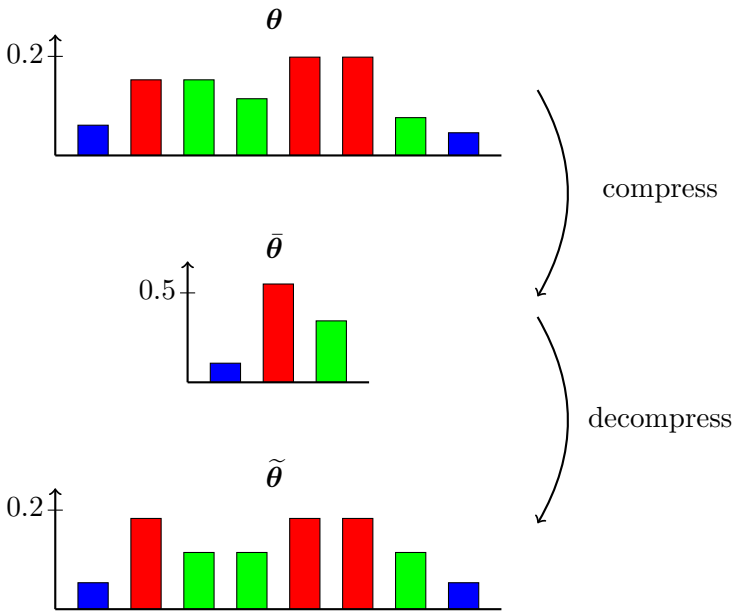


Figure 5.8: The Godzilla process as used by our biclustering model. The cluster assignments are encoded by the colors. Note that the distribution among the same colored elements of $\widetilde{\boldsymbol{\theta}}$ is flat, which was not yet the case for $\boldsymbol{\theta}$.

We are now ready to introduce a different perspective of the IABM: we use a Chinese restaurant process prior for both clusterings. Furthermore, we assume a standard infinite mixture model for the objects which is additionally augmented by a feature clustering that compresses the object cluster components as discussed above. The model is shown in Figure 5.9.

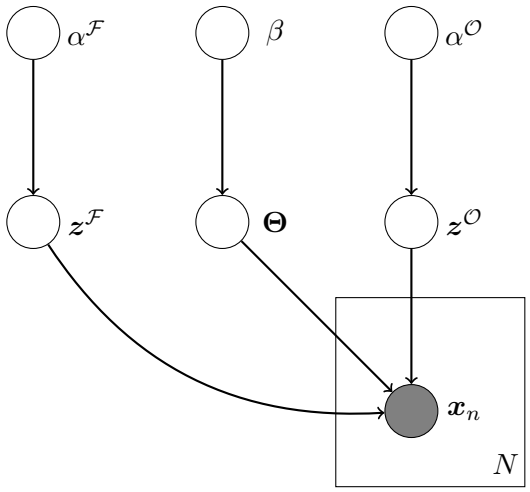


Figure 5.9: Infinite asymmetric biclustering model with the Godzilla process.

The nonparametric generative process for the data looks as follows:

1. Draw a feature clustering $\boldsymbol{z}^{\mathcal{F}}$: $\boldsymbol{z}^{\mathcal{F}} \,|\, \alpha^{\mathcal{F}} \sim \mathrm{CRP}(\alpha^{\mathcal{F}})$.
2. Draw $\boldsymbol{\theta}_\mu \,|\, \beta \sim G_0(\boldsymbol{\lambda})$ for $\mu = 1, 2, \ldots$
3. Draw an object clustering $\boldsymbol{z}^{\mathcal{O}}$: $\boldsymbol{z}^{\mathcal{O}} \,|\, \alpha^{\mathcal{O}} \sim \mathrm{CRP}(\alpha^{\mathcal{O}})$.
4. For the n-th object: Draw $\boldsymbol{x}_n \,|\, z_n^{\mathcal{O}}, \boldsymbol{z}^{\mathcal{F}}, \boldsymbol{\Theta} \sim \mathrm{Mult}(\boldsymbol{x}_n | \widetilde{\boldsymbol{\theta}}_{z_n^{\mathcal{O}}})$.

Here we collected all the cluster components in the (infinite) matrix $\boldsymbol{\Theta}$ and $\lambda_j = \beta$ for $j = 1, \ldots, d$.

Theorem 5.2. *The process given above is equivalent to the IABM in Figure 5.7.*

Proof. The distribution of the clusterings $\boldsymbol{z}^{\mathcal{O}}$ and $\boldsymbol{z}^{\mathcal{F}}$ is the same and thus the distribution of the $x_{i(r)}$ is the same. Let's switch the notation to histograms $\boldsymbol{x}_n$ and consider the likelihood of the data, conditioned on the latent variables. For the IABM in Figure 5.7 we get:

$$P(\boldsymbol{x}_n | \boldsymbol{z}^{\mathcal{O}}, \boldsymbol{z}^{\mathcal{F}}, \{\boldsymbol{\pi}_\mu\}_{\mu=1}^{\infty}) = \mathrm{Mult}(\boldsymbol{x}_n | \boldsymbol{\phi}, M),$$

where $\phi_j = \pi_{z_n^{\mathcal{O}}, z_j^{\mathcal{F}}} / m^{\mathcal{F}}_{z_j^{\mathcal{F}}}$ is the probability of bin j, which is given by the $z_j^{\mathcal{F}}$-th entry of the cluster component to which histogram $\boldsymbol{x}_n$ is assigned, divided by the number of

features assigned to this feature cluster. The likelihood for the model in Figure 5.9 is given by:

$$P(\boldsymbol{x}_n|\boldsymbol{\Theta}, \boldsymbol{z}^{\mathcal{O}}, \boldsymbol{z}^{\mathcal{F}}) = \text{Mult}(\boldsymbol{x}_n|\widetilde{\boldsymbol{\theta}}_{z_n^{\mathcal{O}}}, M),$$

where $\widetilde{\theta}_{z_n^{\mathcal{O}},j} = \bar{\theta}_{z_n^{\mathcal{O}},z_j^{\mathcal{F}}}/m_{z_j^{\mathcal{F}}}^{\mathcal{F}}$ for $j = 1, \ldots, d$, is the cluster component $\boldsymbol{\theta}_{z_n^{\mathcal{O}}}$ perturbed by the Godzilla process. We thus only need to show that $\bar{\boldsymbol{\theta}}_{z_n^{\mathcal{O}}}$ and $\boldsymbol{\pi}_{z_n^{\mathcal{O}}}^{\mathcal{F}}$ have the same distribution, the distribution is independent of the object assignment. We have

$$\bar{\boldsymbol{\theta}} \sim \text{Dir}(\bar{\boldsymbol{\theta}}|\boldsymbol{\lambda}'),$$

and

$$\boldsymbol{\pi} \sim \text{Dir}(\boldsymbol{\pi}|\boldsymbol{\lambda}'),$$

with $\lambda'_\nu = \beta m_\nu^{\mathcal{F}}$. Which proves our claim. For $\bar{\boldsymbol{\theta}}$, we used the partition property of the Dirichlet distribution: let $\mathcal{A}_1, \ldots, \mathcal{A}_k$ be a partition of the domain $\mathcal{X} = \{1, 2, \ldots, N_C^{\mathcal{F}}\}$ then we have:

$$(\theta_{\mathcal{A}_1}, \ldots, \theta_{\mathcal{A}_k}) \sim \text{Dir}(\beta(\mathcal{A}_1), \ldots, \beta(\mathcal{A}_k))$$

where $\beta(\mathcal{A}_i) = \sum_{x_j \in \mathcal{A}_i} \beta_i$ for $j = 1, \ldots, N_C^{\mathcal{F}}$; in our case we use a uniform prior β and thus we get $\beta(\mathcal{A}_i) = \sum_{x_j \in \mathcal{A}_i} \beta$ □

In the Godzilla-IABM representation above, it is now straigthforward to implement a collapsed Gibbs sampler.

5.4 The infinite symmetric biclustering model and the Godzilla process

In this section we discuss a symmetric biclustering model better suited for Gibbs sampling which shows to be equivalent to the ISBM, we will again use the Godzilla process. Let's consider the following generative process:

$$\begin{aligned} \boldsymbol{z}^{\mathcal{O}} \,|\, \alpha^{\mathcal{O}} &\sim \text{CRP}(\alpha^{\mathcal{O}}) \\ \boldsymbol{z}^{\mathcal{F}} \,|\, \alpha^{\mathcal{F}} &\sim \text{CRP}(\alpha^{\mathcal{F}}) \\ \boldsymbol{\Theta} \,|\, \beta &\sim \text{Dir}(\boldsymbol{\Lambda}) \\ \boldsymbol{X} \,|\, \boldsymbol{\Theta}, \boldsymbol{z}^{\mathcal{F}}, \boldsymbol{z}^{\mathcal{O}} &\sim \text{Mult}(\widetilde{\boldsymbol{\Theta}}, R) \end{aligned} \tag{5.5}$$

Here $\widetilde{\Theta}_{i,j} = \bar{\Theta}_{z_i^{\mathcal{O}},z_j^{\mathcal{F}}}/(m_{z_i^{\mathcal{O}}} m_{z_j^{\mathcal{F}}})$, $\bar{\Theta}_{\mu,\nu} = \sum_{i \in \mathcal{O}_\mu} \sum_{j \in \mathcal{F}_\nu} \Theta_{i,j}$ and $\Lambda_{i,j} = \beta$. The Dirichlet and Multinomial distribution are assumed to be defined on matrices, this is equivalent to vectorizing the matrices. The prior $\boldsymbol{\Lambda}$, as well as $\boldsymbol{\Theta}$ have dimension $N \times d$, $\bar{\boldsymbol{\Theta}}$ has dimension $N_C^{\mathcal{O}} \times N_C^{\mathcal{F}}$ and $\widetilde{\boldsymbol{\Theta}}$ has again dimension $N \times d$. As in the previous section, we use again the Godzilla process, but here in 2D: We perturb the Multinomial parameter $\boldsymbol{\Theta}$ for the entire matrix $\boldsymbol{X}$ by the two clusterings. The difference being, that here we use this idea for the generation of $\boldsymbol{X}$, while before in the asymmetric case we only used the feature clustering in this way and for the object clustering we assumed a standard infinite mixture model. The process is shown in Figure 5.10.

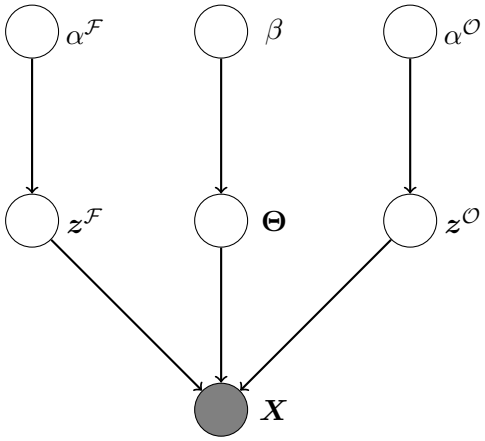


Figure 5.10: Infinite symmetric biclustering model with the Godzilla process.

Theorem 5.3. *Model (5.5) is equivalent to the ISBM in (5.1).*

Proof. In the ISBM the likelihood of the data conditioned on $\boldsymbol{z}^{\mathcal{O}}$, $\boldsymbol{z}^{\mathcal{F}}$ and $\boldsymbol{\Pi}$ is multinomially distributed:

$$P(\boldsymbol{X}|\boldsymbol{z}^{\mathcal{O}}, \boldsymbol{z}^{\mathcal{F}}, \boldsymbol{\Pi}) = \text{Mult}(\boldsymbol{X}|\boldsymbol{\Phi}, R),$$

where $\Phi_{i,j} = \Pi_{z_i^{\mathcal{O}}, z_j^{\mathcal{F}}}/(m^{\mathcal{O}}_{z_i^{\mathcal{O}}} m^{\mathcal{F}}_{z_j^{\mathcal{F}}})$. Similarly, for the model in (5.5) the likelihood of the data is given by:

$$P(\boldsymbol{X}|\boldsymbol{z}^{\mathcal{O}}, \boldsymbol{z}^{\mathcal{F}}, \boldsymbol{\Theta}) = \text{Mult}(\boldsymbol{X}|\widetilde{\boldsymbol{\Theta}}, R),$$

where $\widetilde{\Theta}_{i,j} = \bar{\Theta}_{z_i^{\mathcal{O}} z_j^{\mathcal{F}}}/(m_{z_i^{\mathcal{O}}} m_{z_j^{\mathcal{F}}})$ and $\bar{\Theta} = \sum_{i \in \mathcal{O}_\mu} \sum_{j \in \mathcal{F}_\nu} \Theta_{i,j}$. We thus only need to show that $\bar{\boldsymbol{\Theta}}$ and $\boldsymbol{\Pi}$ have the same distribution, as the two clusterings $\boldsymbol{z}^{\mathcal{O}}$ and $\boldsymbol{z}^{\mathcal{F}}$ have the same distribution in both models. The distribution of $\bar{\boldsymbol{\Theta}}$ is given by:

$$\bar{\boldsymbol{\Theta}} \sim \text{Dir}(\bar{\boldsymbol{\Theta}}|\boldsymbol{\Lambda}'),$$

and

$$\boldsymbol{\Pi} \sim \text{Dir}(\boldsymbol{\Pi}|\boldsymbol{\Lambda}'),$$

where $\Lambda'_{\mu,\nu} = \beta m^{\mathcal{O}}_{\mu} m^{\mathcal{F}}_{\nu}$. Which proves our claim. We used the partition property of the Dirichlet distribution as before in the asymmetric case. □

We now compute the probabilities needed for a Gibbs sampler for the model above. Let's consider assigning the i-th object to object cluster μ, the probability is given by:

$$P(z_i^{\mathcal{O}} = \mu|\boldsymbol{\Theta}, \boldsymbol{X}, \boldsymbol{z}^{\mathcal{O}}_{-i}, \boldsymbol{z}^{\mathcal{F}}) = \prod_{k=1}^{N} \prod_{j=1}^{d} \left(\widetilde{\Theta}_{k,j|z_i^{\mathcal{O}}=\mu, \boldsymbol{z}^{\mathcal{O}}_{-i}, \boldsymbol{z}^{\mathcal{F}}, \boldsymbol{\Theta}} \right)^{X_{k,j}},$$

where $\widetilde{\Theta}_{k,j|z_i^{\mathcal{O}}=\mu, \boldsymbol{z}^{\mathcal{O}}_{-i}, \boldsymbol{z}^{\mathcal{F}}, \boldsymbol{\Theta}}$ is the probability of bin (k, j) after applying the Godzilla process to a given object and feature clustering, specified by $z_i^{\mathcal{O}} = \mu, \boldsymbol{z}^{\mathcal{O}}_{-i}$ and $\boldsymbol{z}^{\mathcal{F}}$ and the

unperturbed parameter $\boldsymbol{\Theta}$. Equivalently for assigning the j-th feature to feature cluster ν:

$$P(z_j^{\mathcal{F}} = \nu | \boldsymbol{\Theta}, \boldsymbol{X}, \boldsymbol{z}_{-j}^{\mathcal{F}}, \boldsymbol{z}^{\mathcal{O}}) = \prod_{i=1}^{N} \prod_{k=1}^{d} \left(\widetilde{\Theta}_{i,k|z_j^{\mathcal{F}}=\nu, \boldsymbol{z}_{-j}^{\mathcal{F}}, \boldsymbol{z}^{\mathcal{O}}, \boldsymbol{\Theta}} \right)^{X_{i,k}}.$$

Note that these probabilities are dependent on all the counts of $\boldsymbol{X}$, which is possibly very large. However, in an implementation, with an appropriate caching of the probabilities involved, one does not need to recompute everything from scratch, as only few terms change.

This still assumes that we know the probabilities of the different bins, which are given by $\boldsymbol{\Theta}$, however, in an actual implementation this needs to be inferred from the data, too. We can compute a MAP estimate as follows:

$$\widetilde{\Theta}_{i,j}^{\text{MAP}} = \bar{\Theta}_{z_i^{\mathcal{O}}, z_j^{\mathcal{F}}}^{\text{MAP}} / (m_{z_i^{\mathcal{O}}}^{\mathcal{O}} m_{z_j^{\mathcal{F}}}^{\mathcal{F}}),$$

for the Godzilla-compressed bins with

$$\bar{\Theta}_{\mu,\nu}^{\text{MAP}} \propto \sum_{i \in \mathcal{O}_\mu} \sum_{j \in \mathcal{F}_\nu} X_{i,j} + \Lambda_{i,j}.$$

In a Gibbs sampler for the inference, we can go through the objects and features in turn and sample the assignments according to the probabilities as given above. In addition to the probabilities for assigning an object or feature to an already existing cluster, we also compute the probability of the data, assuming the object/feature is placed in its own cluster. This probability is then weighted with the concentration parameter of the Dirichlet process, while the other probabilities are weighted by the number of objects/features assigned to the other clusters, just like in the standard Dirichlet process.

5.5 Evaluation

In this section we evaluate the IABM and ISBM as introduced in the preceding sections. For the asymmetric algorithm we preprocessed all of our data, such that we have the approximately same number of counts per object. For the symmetric algorithm we used a different preprocessing step: we "normalize" the entire matrix, such that it contains a fixed total count. As one can already see, we potentially destroy a lot of data in the asymmetric case, as we might completely change the between-objects statistics. The algorithms were, if not stated otherwise, initialized with 5 object clusters and d feature clusters, where d is the number of features.

5.5.1 Osherson dataset – A giant panda swimming in the arctic ocean?

In analogy to [Kemp et al., 2006], we use the animal-feature matrix from [Osherson et al., 1991]. This data has dimension 50×85 and was collected by a psychological experiment: different animals (Siamese cat, killer whale) are given feature ratings (arctic, eats meat)

on a scale from 0 to 100. The goal is then to identify animal clusters (e.g. humpback whale and killer whale should be in the same cluster) and feature clusters (e.g. ocean and water should be in the same cluster).

This dataset is interesting for two reasons: First, the data allows for an easy and intuitive understanding of the result and second, it is real data and not synthetic data. The data set is however not well suited for a thorough analysis, as we are not given a ground truth clustering.
Charles Kemp kindly forwarded us the original data. In their paper [Kemp et al., 2006] they use a thresholded version of the data to get a binary data matrix. As both, our asymmetric and symmetric biclustering algorithms are designed for count data we used the original data, which was additionally preprocessed as discussed above.

Both algorithms consistently identified 11 or 12 object clusters, we show the results of a test run in Figure 5.11 and Figure 5.12, respectively. The model order selection is shown in Figure 5.13.

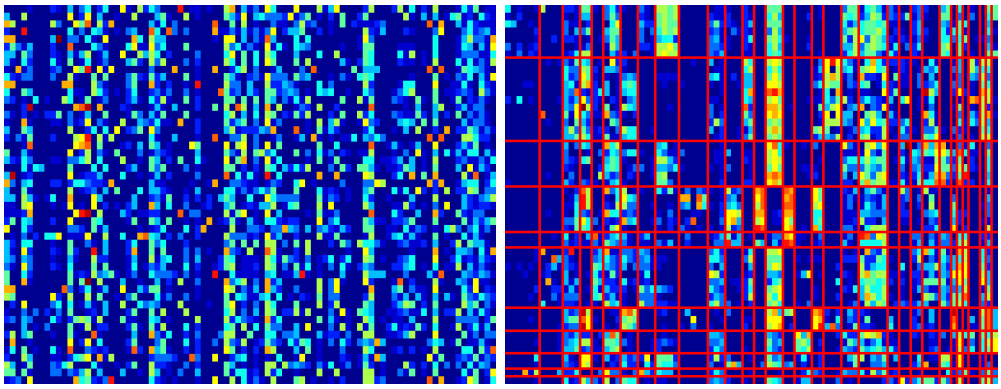


Figure 5.11: Asymmetric biclustering (IABM) of the Osherson data set: input to the left and output to the right.

The number of object clusters does not change a lot for the algorithms after a few iterations of the Gibbs sampler, this is different for the feature clustering, which is more dynamic and changes more frequently. Two interesting object clusters are apparently the “pig” and “antelope” cluster: they were sometimes separated by the algorithms and sometimes merged. In general we got the impression that the object clustering of the symmetric algorithm is slightly inferior.

The partitioning of the animals, as returned by the two algorithms is shown in Table 5.2. The results mostly match the few results published in [Kemp et al., 2006] and both, the symmetric and asymmetric algorithms, give similar results. This however, also largely depends on the concentration parameters, which were set such that the two algorithms, return similar clusterings of the objects.

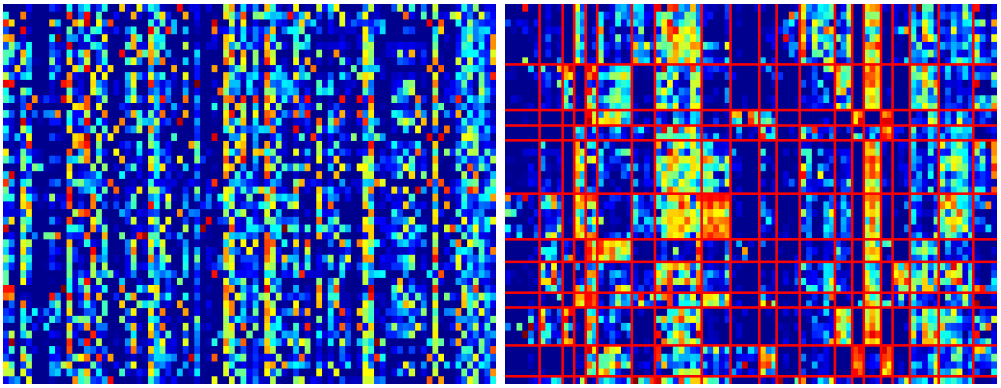
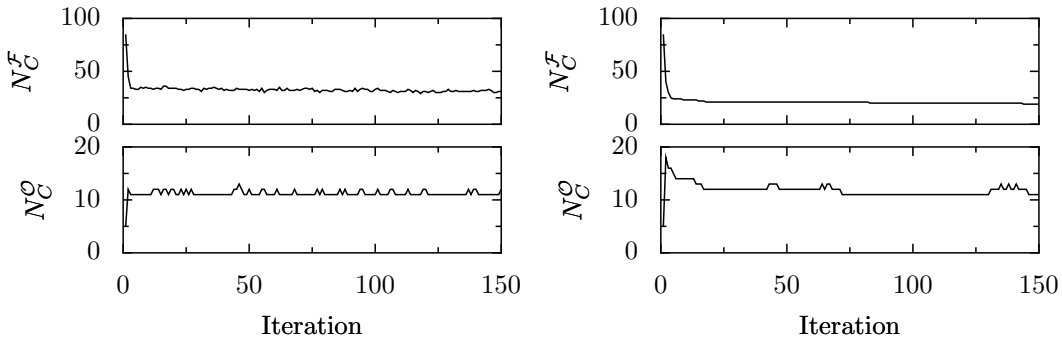


Figure 5.12: Symmetric biclustering (ISBM) of the Osherson data set: input to the left and output to the right.



(a) asymmetric algorithm (IABM)

(b) symmetric algorithm (ISBM)

Figure 5.13: Number of object clusters $N_C^{\mathcal{O}}$ and feature clusters $N_C^{\mathcal{F}}$ as inferred by a Gibbs sampler for the two nonparametric models.

cluster	asymmetric	symmetric
1	tiger, leopard, fox, wolf, *weasel*, bobcat, lion	tiger, leopard, fox, wolf, bobcat, lion
2	antelope, horse, *moose*, *ox*, *sheep*, giraffe, *buffalo*, zebra, deer, pig, cow	antelope, horse, giraffe, zebra, deer
3	dalmatian, Persian cat, German shepherd, Siamese cat, chihuahua, collie	dalmatian, Persian cat, German shepherd, Siamese cat, chihuahua, *weasel*, collie
4	killer whale, *blue whale*, *humpback whale*, seal, walrus, dolphin	killer whale, seal, walrus, dolphin
5	beaver, otter	beaver, otter
6	skunk, mole, hamster, squirrel, rabbit, rat, mouse, raccoon	skunk, mole, hamster, squirrel, rabbit, rat, mouse, raccoon
7	hippopotamus, rhinoceros, elephant	hippopotamus, rhinoceros, elephant
8	spider monkey, gorilla, chimpanzee	spider monkey, gorilla, chimpanzee, *giant panda*
9	grizzly bear, polar bear	grizzly bear, polar bear
10	bat	bat
11	*giant panda*	*moose*, *ox*, *sheep*, *buffalo*, *pig*, *cow*
12		*blue whale*, *humpback whale*

Table 5.2: The two animal clusterings inferred by the IABM and ISBM. The clusters are reordered, to show the overlap. Differences are shown in italic.

cluster	asymmetric	symmetric
1	blue, strainteeth, tusks, plankton, skimmer	blue, strainteeth, tusks, plankton, skimmer
2	fish, arctic, coastal	fish, arctic, coastal
3	flippers, ocean	flippers, ocean
4	big, strong	big, strong
5	swims, water	swims, water
6	hands, bipedal, tree	hands, bipedal, tree
7	walks, quadrapedal, ground	walks, quadrapedal, ground
8	paws, claws	paws, claws
9	orange, red, yellow, *flys*, desert, *cave*	orange, red, yellow, *stripes*, *longneck*, desert
10	hooves, horns, *grazer*	hooves, horns
11	meatteeth, meat, hunter, stalker	meatteeth, meat, hunter, stalker, *fierce*
12	nocturnal, hibernate, scavenger	nocturnal, hibernate, scavenger, *cave*
13	chewteeth, smelly, group	chewteeth, smelly, *vegetation*, *timid*, group
14	brown, tail, fast, active, newworld	*black*, brown, *furry*, tail, fast, active, newworld, *oldworld*
15	buckteeth, weak	*small*, *pads*, buckteeth, weak, *forager*, *forest*
16	hairless, toughskin	*gray*, hairless, toughskin, *bulbous*, *slow*, *inactive*
17	plains, fields	*grazer*, plains, fields
18	*stripes*, hops, tunnels, insects	*flys*, hops, tunnels, insects
19	bush, mountains	*longleg*, bush, *jungle*, mountains
20	lean, muscle, *fierce*	lean, muscle, *agility*, *smart*, *solitary*, *nestspot*
21	patches, spots	*white*, patches, spots, *domestic*

Table 5.3: The two feature clusterings inferred by the IABM and ISBM. The clusters are reordered, to show the overlap. Differences are shown in italic; we did not include the additional clusters of the asymmetric algorithm because of space constraints.

5.5.2 Toy example 1 – "Step" pattern

As a first toy example we consider a dataset with 6 object clusters, each containing 60 objects, and 10 feature clusters, each containing 5 features. The data was created as follows: we define a probability vector for the first object cluster over the 10 feature clusters and use this vector for the other object clusters, where we however change the element positions to get a steps-pattern. We then sequentially sample the objects by assigning them to a cluster and sample a histogram according to its cluster component. In the end we randomly permute the columns and rows of the data matrix we generated. We initialized both of the algorithms with 1 object cluster and with d feature clusters.

The results of both algorithms are shown in Figure 5.14 and the development of the number of clusters in Figure 5.15.

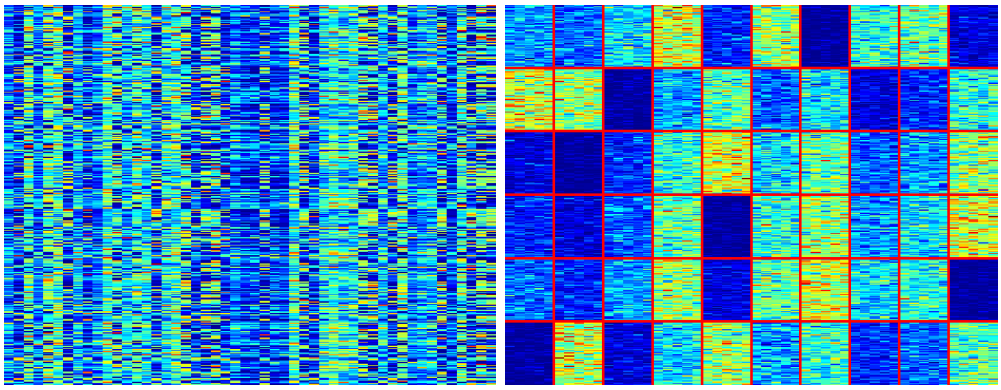
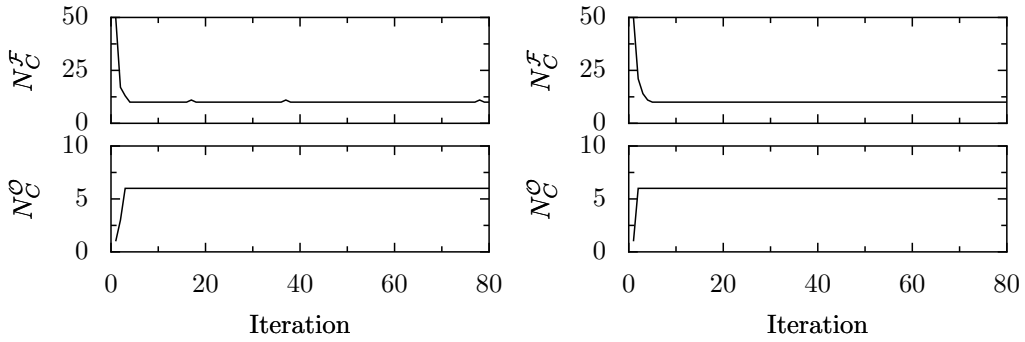


Figure 5.14: Biclustering of the "steps" data set: input to the left and output to the right.



(a) asymmetric algorithm (IABM)

(b) symmetric algorithm (ISBM)

Figure 5.15: Gibbs sampler for the biclustering of the "steps" data set: the development of the number of object clusters $N_C^{\mathcal{O}}$ and number of feature clusters $N_C^{\mathcal{F}}$.

Both algorithms converge very fast and result in the same (correct) biclustering. The data set is inherently asymmetric and thus it surprises that the symmetric algorithm performs so well.

5.5.3 Toy example 2 – Normalization

As a next experiment we consider how the algorithms perform on a synthetic data set, which is special in that two feature clusters have the same distribution over the objects (with probability 0.9 it is from object cluster one, with probability 0.1 from object cluster two). Unsophisticated algorithms might merge the two clusters for this reason, which however does not happen with our algorithms. The data was again asymmetrical generated. See Figure 5.16 and Figure 5.17 for the results.

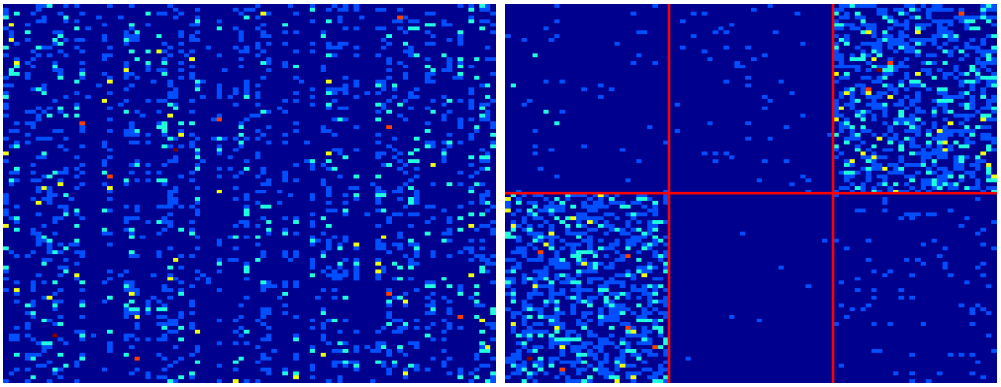
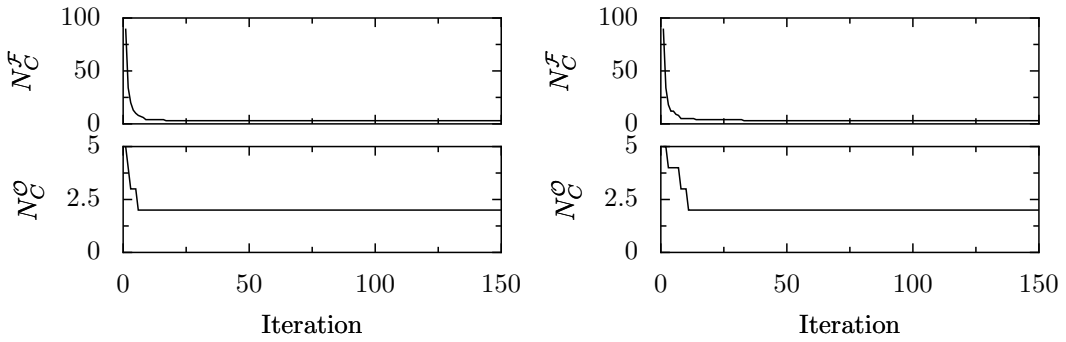


Figure 5.16: Biclustering of the normalization problem data set: input to the left and output to the right.



(a) asymmetric algorithm (IABM)

(b) symmetric algorithm (ISBM)

Figure 5.17: Biclustering of the normalization problem data set: the development of the number of object clusters $N_C^{\mathcal{O}}$ and number of feature clusters $N_C^{\mathcal{F}}$.

Again, both algorithms converge within a few iterations to the correct solution.

5.5.4 Data sampled from the symmetric biclustering model

So far all of the data sets were essentially asymmetric. Here we now consider data sampled from the symmetric biclustering model. We would expect the symmetric algorithm to perform better than the asymmetric one. For the generation of the synthetic data, we first generate an object and feature clustering and then sample a discrete distribution over the just generated biclusters from a Dirichlet distribution.

One problem is now the computation of the error of the inferred biclustering for a comparison: although we know the ground-truth biclustering; for a small prior belief many of these biclusters will have a probability $\Pi_{\mu,\nu}$ of almost zero. It might thus very well happen that almost all draws are only from one bicluster, which our algorithm will identify, but it will also merge all the "zero bins" into as few biclusters as possible. Computing the assignment error of the inferred biclustering compared to the ground-truth with the Hungarian method will possibly result in a large error, because there is no way for our algorithm to find a difference between the bins as all have zero counts. As a simple heuristic we also computed the assignment error of the inferred biclustering compared to a biclustering with all the zero rows/columns in a separate cluster as the ground-truth. We then chose the error as the minimum of the two errors. This heuristic works well for really small prior beliefs, but seems to introduce some artifacts for moderately small prior beliefs, as first additional biclusters evolve and thus have nonzero counts, but our biclustering algorithm will still decide to merge these bins with the zero biclusters resulting in an error. The problem is illustrated in Figure 5.18.

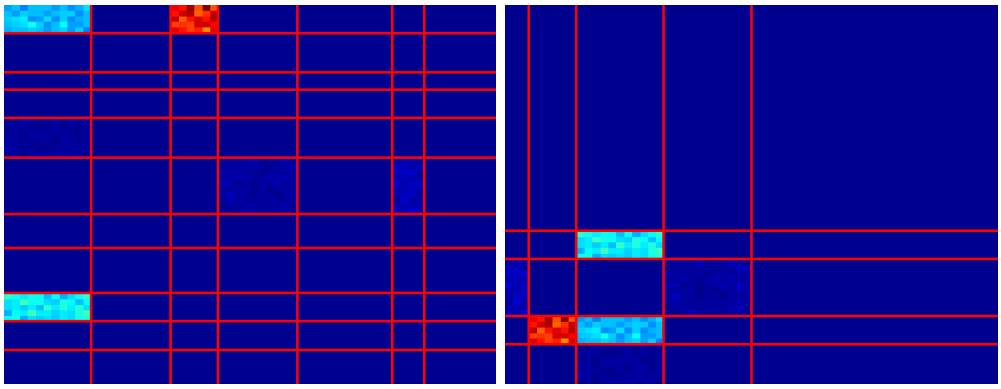


Figure 5.18: The problem with the assignment error computation: To the left in the ground-truth biclustering we have many biclusters with zero counts, to the right in the inferred biclustering they are merged into larger biclusters. The biclustering to the right would have a large error, although it is arguably a good biclustering.

The results of the ISBM are shown in Figure 5.19. As we can see the error is mostly significantly under 10%, except for very "flat" biclustering configurations, i.e. a uniform distribution among the biclusters, which we get for large priors. Our algorithm then

returns one bicluster spanning over the whole matrix, which is arguably the best it can do, as there is apparently no difference in the co-occurrence in the object/feature pairs for such data, see Figure 5.20 for an example.

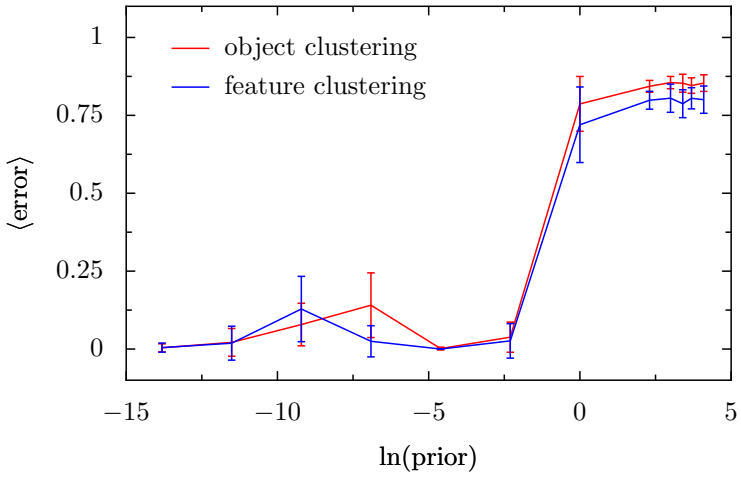


Figure 5.19: Biclustering of synthetic data from the symmetric biclustering model. The "bump" around -8 is at least in part a consequence of the validation problem discussed above.

When we run the asymmetric algorithm on this data set we see that it works well in the prior range around −5. Non-surprisingly it results in one bicluster too for very uniform biclusters. The asymmetric biclustering algorithm fails however for the very peaked biclustering configurations when the log of the prior is smaller than −10, as it is clearly not designed for such data.

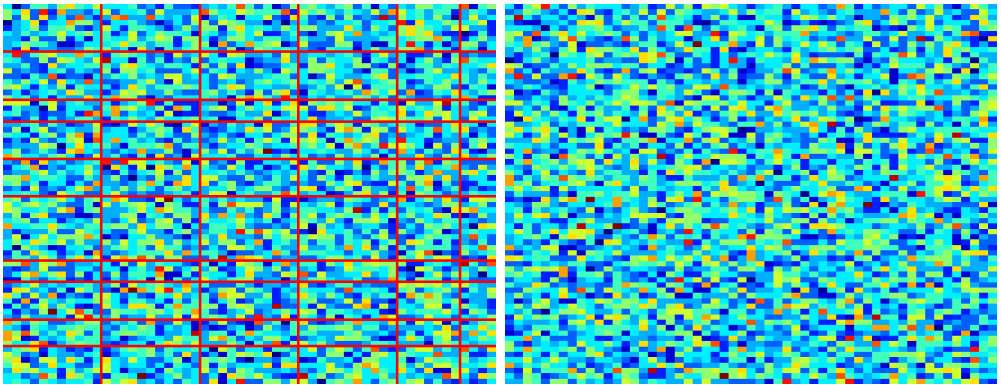


Figure 5.20: Ground-truth biclustering to the left and a solution found by our algorithm (only one bicluster).

5.5.5 More synthetic data

Here, we consider one of the synthetic data sets used in [Prelić et al., 2006]. In Figure 5.21 we show two biclusterings of the noisy data sets as returned by the symmetric biclustering algorithm.

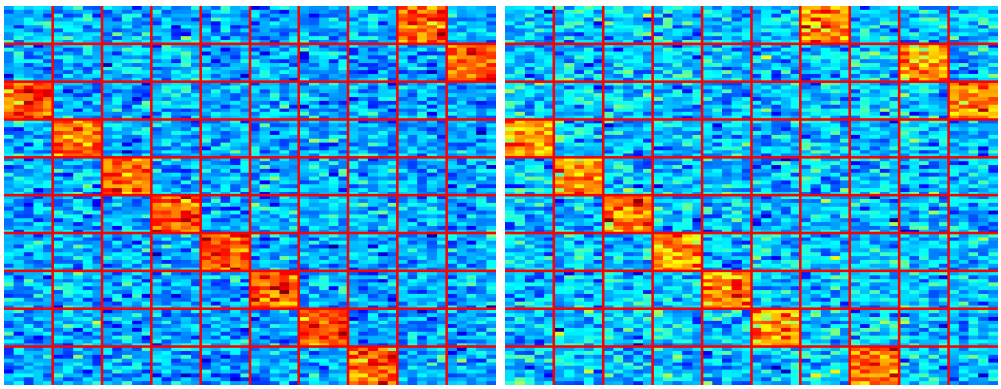


Figure 5.21: Biclusterings of the noisy data set obtained by the symmetric algorithm.

The symmetric algorithm identifies almost always all of the clusters correctly, as can be seen in Figure 5.22. We used the same $\alpha^{\mathcal{F}}$ and $\alpha^{\mathcal{O}}$ for all of the noisy data sets of scenario 1.

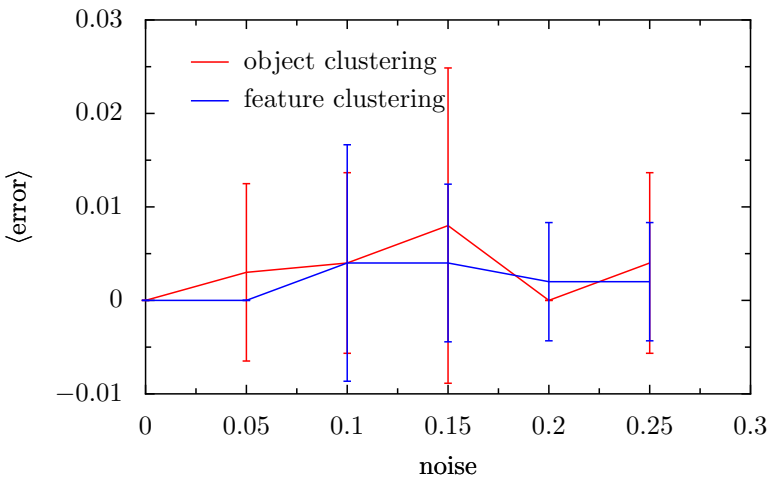


Figure 5.22: Average error rate of the two clusterings computed by the symmetric algorithm for the noise data set.

5.6 Discussion, open problems and future directions

In this chapter we introduced a symmetric (ISBM) and an asymmetric (IABM) biclustering model. While the asymmetric model showed to lead to good results on some toy data sets and might be especially interesting for unsupervised feature selection, the symmetric algorithm does arguably capture the biclustering idea better: it handles both, the objects and features, in a completely symmetric way and it also copes well with data where only a few objects and features show co-occurrence. Our two biclustering algorithms are nonparametric and do not assume the number of biclusters is known a-priori.

Selecting good parameters $\alpha^{\mathcal{O}}, \alpha^{\mathcal{F}}$ and β for a given data set still involves parameter tuning performed by a human. As the prior and the two concentration parameters measure in parts similar aspects of the biclustering (a small prior β leads to only few biclusters as many biclusters have probability zero and are merged) it would be interesting to see, whether we could constraint the parameters.

The results of our nonparametric algorithms showed to be promising, but further analysis and experiments are needed to give a quantitative measure of its prediction performance. Also, a comparison of our symmetric algorithm to the IRM and the FIRM is needed for identifying the weaknesses and strengths of the models; a comparison with other non-Dirichlet process based methods would also be highly desirable. We would expect our model to perform better than the IRM, as it also takes into account the frequency of a co-occurrence, just like the FIRM; a comparison with the FIRM is harder, as both methods employ different approaches for handling non-binary data. Our approach is equivalent to an infinite mixture model for the biclusters, allowing for a non-factorized distribution of the biclusters, and for each draw really generates a count in the data matrix. In the FIRM model the data is explained by a factorized Multinomial distribution for the objects and features and a Bernoulli draw that decides whether to generate a count; like this negative counts are possible which essentially also results in a non-factorized Multinomial model. We feel that our model is conceptually nicer, however that being said, so far we were not yet able to derive a variational algorithm for the inference, which was given for the FIRM. Variational methods would be needed for applying our algorithm to large-scale data sets, such as natural language processing or microarray data.

Bibliography

Hirotugu Akaike. A new look at the statistical model identification. *IEEE Transactions on automatic control*, 19:716–723, 1974.

C. E. Antoniak. Mixtures of Dirichlet processes with applications to nonparametric problems. *The Annals of Statistics*, 2:1152–1174, 1974.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. 2007.

David Blackwell and James MacQueen. Ferguson distributions via pólya urn schemes. *The Annals of Statistics*, 1:353–355, 1973.

David M. Blei and Michael I. Jordan. Variational inference for Dirichlet process mixtures. *Journal of Bayesian Analysis*, 1(1):121–144, 2005.

Hans-Hermann Bock. Two-way clustering for contingency tables: Maximizing a dependence measure. In M. Schader, W. Gaul, and M. Vichi, editors, *Between data science and applied data analysis*, pages 143–154. 2003.

Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analalyis and Machine Intelligence*, 23(11):1222–1239, 2001.

Christopher A. Bush and Steven N. MacEachern. A semiparametric Bayesian model for randomised block designs. *Biometrika*, 83:275–355, 1996.

Stanislav Busygin, Oleg A. Prokopyev, and Panos M. Pardalos. Feature selection for consistent biclustering via fractional 0-1 programming. *Journal of Combinatorial Optimization*, 10(1):7–21, 2005.

Yizong Cheng and George M. Church. Biclustering of expression data. In *Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93–103, 2000.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(Series B):1–38, 1977.

Inderjit S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Knowledge Discovery and Data Mining*, pages 269–274, 2001.

Michael D. Escobar. Estimating normal means with a Dirichlet process prior. *Journal of the American Statistical Association*, 89:268–277, 1994.

Michael D. Escobar and Mike West. Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90:577–588, 1995.

Thomas S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

M. R. Garey and D. S. Johnson. *Computers and Intractability: A guide to the theory of NP-completeness*. 1979.

Zoubin Ghahramani and Matthew J. Beal. Propagation algorithms for variational bayesian learning. In *Advances in Neural Information Processing Systems*, pages 507–513, 2001.

Tom Griffiths and Zoubin Ghahramani. Infinite latent feature models and the Indian buffet process. In *Advances in Neural Information Processing Systems*, 2005.

Thomas Hofmann and Jan Puzicha. Statistical models for co-occurrence data. Technical report, Massachusetts Institute of Technology, 1998.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 37(2): 183–233, 1999.

Charles Kemp, Joshua B. Tenenbaum, Thomas L. Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. In *National Conference on Artificial Intelligence*, 2006.

Harold W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, (2):83–97, 1955.

Kenichi Kurihara, Yoshitaka Kameya, and Taisuke Sato. A frequency-based stochastic blockmodel. In *Workshop on Information-Based Induction Sciences*, 2006.

Kenichi Kurihara, Yoshitaka Kameya, and Taisuke Sato. Discovering concepts from word co-occurrences with a relational model. *Transactions of the Japanese Society for Artificial Intelligence*, 22(2), 2007a.

Kenichi Kurihara, Max Welling, and Yee Whye Teh. Collapsed variational dirichlet process mixture models. In *International Joint Conference on Artificial Intelligence*, pages 2796–2801, Hyderabad, India, 2007b.

Kenichi Kurihara, Max Welling, and Nikos Vlassis. Accelerated variational Dirichlet process mixtures. In Bernhard Schölkopf, John Platt, and Thomas Hoffman, editors, *Advances in Neural Information Processing Systems*, pages 761–768, Cambridge, MA, 2007c. MIT Press.

T. Lange, M. Braun, V. Roth, and J. Buhmann. Stability-based model selection. In *Advances in Neural Information Processing Systems*, 2003.

M. Law, A. Jain, and M. Figueiredo. Feature selection in mixture-based clustering. In *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003. MIT Press.

M. H. Law, M. Figueiredo, and A. K. Jain. Feature saliency in unsupervised learning. Technical report, Department of Computer Science and Engineering, Michigan State University, 2002.

Steven N. MacEachern. Estimating normal means with a conjugate style Dirichlet process prior. *Communications in Statistics: Simulation and Computation*, 23:727–741, 1994.

Sara Madeira and Arlindo Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1: 24–45, 2004.

Edward Meeds, Zoubin Ghahramani, Radford M. Neal, and Sam T. Roweis. Modeling dyadic data with binary latent factors. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, pages 977–984, Cambridge, MA, 2007. MIT Press.

Radford M. Neal. Bayesian mixture modeling. In C. R. Smith, Gary J. Erickson, and Paul O. Neudorfer, editors, *Maximum Entropy and Bayesian Methods: Proceedings of the 11th International Workshop on Maximum Entropy and Bayesian Methods of Statistical Analysis*, pages 197–211, Dordrecht, 1991. Kluwer Academic Publishers.

Radford M. Neal. Probabilisitic inference using Markov chain Monte Carlo methods. Technical report, Department of Computer Science, University of Toronto, 1993.

Radford M. Neal. Markov chain sampling methods for Dirichlet process mixture models. Technical report, Department of Statistics, University of Toronto, 1998.

Daniel N. Osherson, Joshua Stern, Ormond Wilkie, Michael Stob, and Edward E. Smith. Default probability. *Cognitive Science*, 15(2):251–269, 1991.

A. Prelić, S. Bleuler, P. Zimmermann, A. Wille, P. Bühlmann, W. Gruissem, L. Hennig, L. Thiele, and E. Zitzler. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, 22(9):1122–1129, 2006.

Jan Puzicha, Thomas Hofmann, and Joachim M. Buhmann. Histogram clustering for unsupervised segmentation and image retrieval. *Pattern Recognition Letters*, 20(9): 889–909, 1999.

Volker Roth and Tilman Lange. Feature selection in clustering problems. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, MA, 2004. MIT Press.

Mark J. Schervish. *Theory of Statistics*. Springer Series in Statistics, 1995.

Gideon Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2): 461–464, 1978.

Yevgeny Seldin, Noam Slonim, and Naftali Tishby. Information bottleneck for non co-occurrence data. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, pages 1241–1248, Cambridge, MA, 2007. MIT Press.

Jayaram Sethuraman. A constructive definition of Dirichlet priors. *Statistica Sinica*, 4: 639–650, 1994.

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. In *37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on Information Theory*, 49(5):1120–1146, 2003.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. Technical report, Department of Statistics, University of California, Berkeley, 2003.

Mike West. Hyperparameter estimation in Dirichlet process mixture models. Technical report, Institute of Statistics and Decision Sciences, Duke University, 1992.

Mike West, Peter Müller, and Michael D. Escobar. Hierarchical priors and mixture models with applications in regression and density estimation. *Aspects of Uncertainty*, pages 363–386, 1994.

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In *Advances in Neural Information Processing Systems*, pages 689–695, 2000.

Index