Markov Chain Monte Carlo for Bayesian Non-parametric Mixture Models

April 12, 2016

James Thornton
Supervised by Dr Anthony Lee
Student ID: 1204121
Warwick University

Contents

1 Introduction **3**

2 Dirichlet Process **5**
2.1 Dirichlet Distribution . . . 5
2.2 Dirichlet Process . . . 6
2.3 Stick Breaking Construction . . . 7
2.4 Blackwell-MacQueen Urn Scheme . . . 8
2.5 Chinese Restaurant Process . . . 10
2.6 Parameter Interpretation . . . 12

3 Bayesian Inference on Mixture Models **13**
3.1 Finite Mixture Model . . . 13
3.2 Dirichlet Process Mixture Model . . . 14
3.3 Bayesian Inference . . . 15

4 Markov Chain Monte Carlo **17**
4.1 Markov Chains . . . 17
4.2 Metropolis Hastings and Gibbs Sampling . . . 18
4.3 Auxiliary Variables and Cycles of Kernels . . . 19

5 Algorithms **21**
5.1 Algorithm 1: Blackwell MacQueen Urn Scheme . . . 21
5.2 Algorithm 2: Chinese Restaurant Process . . . 24
5.3 Algorithm 3: Marginalised Chinese Restaurant Process . . . 28
5.4 Algorithm 5: Chinese Restaurant Process with Metropolis Hastings Step . . . 30
5.5 Algorithm 6: Urn Scheme with Metropolis Hastings Step . . . 32
5.6 Algorithm 7: Metropolis Hastings with Specific Proposal . . . 33
5.7 Algorithm 8: Auxiliary Variable Method . . . 37

6 Algorithm Evaluation **41**
6.1 Neal's 9 Data Points . . . 41
6.2 2-Dimensional Normal Wishart Prior . . . 43

7 Dirichlet Process Multinomial Logistic Model **45**
7.1 Model Formulation . . . 45
7.2 Experimentation . . . 46
7.2.1 Wine Data . . . 48

8 Hierarchical Dirichlet Process Mixture Model **51**
8.1 Chinese Restaurant Franchise . . . 52
8.2 Topic Modelling . . . 55
8.2.1 Tweet Data . . . 57

9 Conclusion & Future Work **60**

10 Appendices **61**
10.1 Stationarity Proofs . . . 61
10.2 Topic Model Tweets . . . 64
10.3 R-Code . . . 65

11 References **66**

1 Introduction

Non-parametric models are models where the number of parameters is determined by the data, this often provides a degree of robustness associated with fewer assumptions. The Dirichlet Process Mixture Model is a very flexible non-parametric model with attractive mathematical properties and applications in both supervised classification and unsupervised clustering, where the number of clusters is "learnt" from the data.

This dissertation concerns the use of Markov Chain Monte Carlo (MCMC) procedures in performing Bayesian inference on non-parametric mixture models. In particular, this report will focus on Dirichlet Process Mixture Models and the algorithms presented in [3]. Basic MCMC procedures and concepts will be explored such as Gibbs sampling and Metropolis-Hastings. It is hoped that this report will provide an accessible introduction to both MCMC and Dirichlet Process Mixture Models. An aim of this dissertation is to provide a comprehensive explanation of why the algorithms presented in [3] are valid. Another source of originality in this work is to explore how the presented algorithms can be extended to developments of the Dirichlet Process Mixture Model and applied to real data for practical purposes. The Dirichlet Process Multinomial Logistic Model shall be applied to classify Italian wines according to region, and the Hierarchical Dirichlet Process Mixture Model shall be used to perform clustering of "Tweets" from news outlets.

Section 2 will introduce the Dirichlet Process and some useful properties that allows it to be used in mixture models, which shall then be introduced in section 3. Section 4 will give an introduction to the MCMC techniques that will be used, followed by a thorough explanation of algorithms 1-8 (excluding 4) of [3] in section 5. These algorithms have been implemented (in R) and the performance of the algorithms shall be explored in section 6. Section 7 will explain how the Dirichlet Process Mixture Model can be extended into the Dirichlet Process Multinomial Logistic Model, which is used to perform supervised classification. Section 8 will then explore the extension known as the Hierarchical Dirichlet Process Mixture Model and how some of the simpler algorithms (algorithms 2 and 3) can be adapted for this more complex model.

Notation

Density

Although only basic MCMC procedures are performed, the Dirichlet Process involves mixtures of discrete and continuous distributions which are notationally quite cumbersome. In addition, as shall be explained in more detail later, the Dirichlet Process is flexible in that it may be parametrised by a discrete or continuous base distribution. This leads to many notational issues when discussing "probability densities" or "mixture densities". Throughout this dissertation, I have been purposely vague when discussing densities to allow for flexibility. The model primarily has either two or three distributions which shall be labelled F, G_0 and H. Each of these distributions can either be discrete or continuous depending on the modelling choice and data in question. For convenience the density of each shall be given in lower case as f, g_0 and h. If continuous, the "density" refers to the Radon-Nikodym density with respect to the Lebesgue measure. Similarly if discrete, the "density" will again be the Radon-Nikodym density with respect to the counting measure.
Therefore, for any integrable function, φ, the following shall be used:

$$\int \varphi(x) dG_0(x) = \int \varphi(x) g_0(x) dx$$

Where "dx" is used for both the counting measure and Lebesgue measure depending on choice of g_0.

Consider a mixture of discrete and continuous distributions of the form:

$$X \sim \frac{1}{n+\alpha}\left(\sum_{i=1}^{n} \delta_{x_i} + \alpha G_0\right)$$

where G_0 is a continuous distribution and α is a positive scalar. Simulating X can be interpreted as taking a draw from $\frac{1}{n}\sum_{i=1}^{n}\delta_{x_i}$ with probability $\frac{n}{n+\alpha}$ or taking a draw from G_0 with probability $\frac{\alpha}{n+\alpha}$. For the random variable X on some underlying probability space with measure $\mathbb{P}$ and for measurable set B:

$$\mathbb{P}[X \in B] = \frac{1}{n+\alpha}\left(\sum_{i=1}^{n}\mathbb{I}_B(x_i) + \alpha\int_B g_0(x)dx\right)$$

Indices

Throughout this report, there will be groups of parameters with the same symbol which are identifiable by their index, given as a subscript. In the interest of readability, the following notation will be used. The same notation shall be used for both sets and vectors, it should be clear from context which one is being used.

$$\begin{aligned} x_{1:n} &= \{x_1, x_2, \ldots, x_n\} \\ x_{-i} &= \{x_j | j \neq i\} \\ x_{1:n} &= (x, x_2, \ldots, x_n) \\ x_{-i} &= (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) \end{aligned}$$

In addition, where m is the number of unique values in set $\theta_{1:n}$, m_{-i} is the number of unique values in θ_{-i}. Explicitly:

$$\begin{aligned} m &= |\{\theta_j | j \in \{1, \ldots, n\}\}| \\ m_{-i} &= |\{\theta_j | j \in \{1, \ldots, n\} \backslash \{i\}\}| \end{aligned}$$

2 Dirichlet Process

The Dirichlet Process is defined in this section as a mathematical object independently of its use in mixture models. Key representations of the Dirichlet Process shall be described here, then exploited in later sections of the dissertation to perform Bayesian inference using Markov Chain Monte Carlo techniques. In the interest of accessibility, only key results and their relevance to the Dirichlet Process are included. For the more curious reader, references for proofs have been included. Some interesting properties of the Dirichlet Process are also described as these will motivate the use of the Dirichlet Process in later sections.

2.1 Dirichlet Distribution

The Dirichlet distribution is a continuous probability distribution for a random finite vector, $P = (P_1, P_2, ..., P_K)$, for any positive integer $K > 1$ where where $\sum_{i=1}^{K} P_i = 1$, $P_i \in [0, 1]$ $\forall i$. Formally, the K-dimensional Dirichlet distribution is a probability distribution characterized by the probability density function, f_P:

$$\begin{aligned} P &\sim \text{Dirichlet}(\boldsymbol{\alpha}) \\ f_P((p_1, p_2, ...p_K)|\boldsymbol{\alpha}) &= \frac{\Gamma(\sum_{i=1}^{K} \alpha_i)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \prod_{i=1}^{K} p_i^{\alpha_i - 1} \end{aligned}$$

$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)$ is a vector of K positive real parameters. K can be any positive integer.
A realisation of the vector P can be interpretted as a vector of event probabilities which parametrise the K dimensional Categorical distribution. Given the density above, it can be seen that the Dirichlet distribution is the conjugate prior of the Categorical distribution. This means that if $Z|P = (p_1, \ldots, p_K) \sim \text{Categorical}((p_1, \ldots, p_K))$, and $P \sim \text{Dirichlet}(\boldsymbol{\alpha})$, then given z_i is a realisation of random variable Z , the posterior distribution is distributed according to a Dirichlet distribution, $P|Z = z_i \sim \text{Dirichlet}(\boldsymbol{\alpha'})$.

$$\begin{aligned} \boldsymbol{\alpha'} &= (\alpha'_1, \ldots, \alpha'_K) \\ \alpha'_j &= \begin{cases} \alpha_j & \text{if } j \neq i \\ \alpha_j + 1 & \text{if } j = i \end{cases} \end{aligned}$$

.
Explicitly, by Bayes' Theorem $f_{P|Z=z_i}(\boldsymbol{p}) \propto \mathbb{P}[Z = z_i|P = (p_1, \ldots, p_K)] f_P((p_1, p_2, ...p_K)|\boldsymbol{\alpha})$, then using $\mathbb{P}[Z = z_i|P = (p_1, \ldots, p_k)] = p_i$, and f_P above:

$$f_{P|Z=z_i}(\boldsymbol{p}) \propto p_i \prod_{j:j\neq i} p_j^{\alpha_j - 1} = \prod_{j=1}^{K} p_j^{\alpha'_j - 1}$$

This is the unnormalised density for the Dirichlet distribution with parameter $\boldsymbol{\alpha'} = (\alpha'_1, \ldots, \alpha'_K)$, detailed above.

Moments of Dirichlet Distribution

The following integral will be used as a building block to show properties of the Dirichlet Process later. This result will also be used in the Hierarchical Dirichlet Process Topic Model in section (8).

Lemma

$$\mathbb{E}[P_1^{r_1} P_{2,}^{r_2} \ldots, P_K^{r_K}] = \frac{\Gamma(\sum_{i=1}^{K} \alpha_i)}{\Gamma(\sum_{i=1}^{K} \alpha_i + R)} \frac{\prod_{i=1}^{K} \Gamma(\alpha_i + r_i)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \tag{1}$$

Where: $P = (P_1, P_2, ...P_K) \sim \text{Dirichlet}((\alpha_1, \alpha_2 \ldots \alpha_K)$, $R = \sum_{i=1}^{K} r_i$

Proof.

$$\mathbb{E}[P_1^{r_1}P_{2,}^{r_2}\ldots,P_K^{r_K}] = \int p_1^{r_1}p_{2,}^{r_2}\ldots,p_K^{r_K}\frac{\Gamma(\sum_{i=1}^K\alpha_i)}{\prod_{i=1}^K\Gamma(\alpha_i)}\prod_i^K p_i^{\alpha_i-1}dp_{1:K} = \int \frac{\Gamma(\sum_{i=1}^K\alpha_i)}{\prod_{i=1}^K\Gamma(\alpha_i)}\prod_{i=1}^K p_i^{\alpha'_i-1}dp_{1:K}$$

where $\alpha'_i = \alpha_i + r_i \ \forall i \in \{1,\ldots K\}$. The integrand is in the same form as a Dirichlet density without the normalising constants, so integration results in the normalising constant:

$$\begin{aligned}\int \frac{\Gamma(\sum_{i=1}^K\alpha_i)}{\prod_{i=1}^K\Gamma(\alpha_i)}\prod_{i=1}^K p_i^{\alpha'_i-1}dp_{1:K} &= \frac{\Gamma(\sum_{i=1}^K\alpha_i)}{\prod_i^k\Gamma(\alpha_i)}\frac{\prod_{i=1}^K\Gamma(\alpha'_i)}{\Gamma(\sum_i^K\alpha'_i)}\\ &= \frac{\Gamma(\sum_{i=1}^K\alpha_i)}{\Gamma(\sum_{i=1}^K\alpha_i+r_i)}\frac{\prod_{i=1}^K\Gamma(\alpha_i+r_i)}{\prod_{i=1}^K\Gamma(\alpha_i)}\end{aligned}$$

□

2.2 Dirichlet Process

The Dirichlet Process, identified by Ferguson in [1], is a generalisation of the Dirichlet distribution to infinite dimension. Let $G \sim DP(\alpha, G_0)$ denote that G is distributed according to a Dirichlet Process (DP), I shall also use the phrase G is a DP.

$$G \sim DP(\alpha, G_0) \tag{2}$$

$$\theta_i|G \overset{iid}{\sim} G,\ i \in \mathbb{N}$$

Consistent with formulation (2), let $\{\theta_i\}_i$ be random variables taking values in set A, with corresponding σ-algebra $\mathcal{A}$, with some unknown probability distribution, $G\in \mathcal{G}$. $\mathcal{G}$ is the set of all probability measures on measurable space $(A, \mathcal{A})$, a common notation for $\mathcal{G}$ is $\mathcal{G} = [0,1]^{\mathcal{A}}$. Let be $\mathcal{B}$ be a σ-algebra on $\mathcal{G}$, e.g. let $B_{A,r} = \{G \in \mathcal{G}|G(A) < r\}$, and $\mathcal{B}$ =$\sigma(\{B_{A,r}|\forall A \in \mathcal{A}, \forall r \in [0.1]\})$. A Dirichlet Process is a particular random variable in $(\mathcal{G}, \mathcal{B})$, which satisfies (3) with parameters α and G_0. Although some proofs such as those of [7] require that the set A be a complete, separable metric space, in general the the Dirichlet Process is a random distribution on an arbitrary measurable space $(A, \mathcal{A})$.

Definition
A random probability measure G, on measurable space $(A, \mathcal{A})$, is distributed according to a Dirichlet Process if for any partition of measurable sets $A_1, ..., A_m$ of A, where m is any positive natural number:

$$(G(A_1), G(A_2), ...G(A_m)) \sim \text{Dirichlet}(\alpha G_0(A_1), \alpha G_0(A_2), ...\alpha G_0(A_m)) \tag{3}$$

α and G_0 are parameters. α is a positive, real valued scalar known as the concentration parameter and G_0 is known as the base distribution over support set A.

Each $G(A_i)$ is a random variable on the same probability space, taking values in $[0, 1]$. This means a DP, G, is a stochastic process indexed by sets $B \in \mathcal{A}$. Kolmogorov's extension theorem can be used to show that there exists a probability distribution, $DP(\alpha, G_0)$ on $(\mathcal{G}, \mathcal{B})$ such that $G \sim DP(\alpha, G_0)$, thus the DP is a well defined random process. The proof is omitted here, but an accessible proof is given by Ferguson (1973) [1], page 214.

It is clear that when support set A is finite, the Dirichlet Process described above simplifies to the Dirichlet distribution. Take the most fine refinement of the finite set A, this would be the set of points that make-up $A = \{a_1, \ldots, a_N\}$. $\forall i \in \{1,\ldots N\}$, let:

$$\begin{aligned}A_i &:= \{a_i\}\\ P_i &:= G(\{a_i\})\\ \alpha_i &:= \alpha G_0(\{a_i\})\end{aligned}$$

Using Ferguson's definition and the definition of the Dirichlet distribution, the Dirichlet Process, G, can be interpretted as a random discrete distribution over support A parametrised by $P = (P_1, \dots P_N) \sim \text{Dirichlet}((\alpha_1, \dots, \alpha_N))$. Hence for the finite case:

$$G(\cdot) = \sum_{i=1}^{N} P_i \delta_{a_i}(\cdot) \tag{4}$$

2.3 Stick Breaking Construction

It may be difficult to get a firm and practical comprehension of the Dirichlet Process from Ferguson's definition (3). However, as shown by (4), the finite-case takes quite an accessible form and this has been extended to the general case by [2] in what is known as the Stick Breaking Construction (SBC). A cornerstone in the SBC is a distribution known as the Griffiths-Engen-McCloskey (GEM), named after its use in genetics [8].
$\boldsymbol{\pi} = (\pi_1, \pi_2, \dots) \sim \text{GEM}(\alpha)$ if $\forall k \in \mathbb{N}$:

$$\begin{aligned} \beta_k &\overset{iid}{\sim} \text{Beta}(1, \alpha) \\ \pi_k &= \beta_k \prod_{j=1}^{k-1} (1 - \beta_j) \end{aligned}$$

There is a well known analogy to interpret the stick breaking construction of the Dirichlet Process. Consider a stick of unit length. A proportion of this stick is snapped away, then a proportion of the remainder is broken away, this snapping procedure continues (countably) infinitely many times. In this construction, the lengths of the sticks broken away from the original stick correspond to probability masses denoted $\{\pi_k\}_{k=1}^{\infty}$. Each probability mass is associated with a value drawn from the base distribution, G_0 and this creates a discrete distribution.

Let $\beta_k \in (0, 1)$ be the $\text{Beta}(1, \alpha)$ distributed random variable corresponding to the proportion snapped from the remaining stick on the k^{th} iteration and π_k correspond to the length snapped off. $\{\pi_k\}_{k=1}^{\infty}$ is a sequence of random probability masses generated as described above. β_k and c_k are random variables on some probability space. β_k, and c_k are each independently identically distributed (iid) and mutually independent.

Define G as:

$$G = \sum_{k=1}^{\infty} \pi_k \delta_{c_k} \tag{5}$$

$$c_k \overset{iid}{\sim} G_0 \tag{6}$$

$$\boldsymbol{\pi} = (\pi_1, \dots, \pi_k) \sim \text{GEM}(\alpha) \tag{7}$$

In [2], Sethuraman proved that the random measure G is a Dirichlet Process, and that any Dirichlet Process takes this form. This construction is a direct way of proving the existence of the Dirichlet Process and its properties, in particular that G is almost surely discrete.

Despite the insights that this construction provides, it is difficult to use the Stick Breaking construction in implementing the Dirichlet Process programatically. Truncating the summation in (5) is one approximation, however this will have a bias. There are however other representations of the distribution $G \sim DP(\alpha, G_0)$, in particular the Blackwell-MacQueen urn scheme and the Chinese Restaurant Process. These representations allow one to take draws from the random Dirichlet Process, G, without explicitly knowing G. These other representations illustrate useful properties of the Dirichlet Process.

2.4 Blackwell-MacQueen Urn Scheme

$$\begin{aligned} \theta_i|G &\overset{iid}{\sim} G \\ G &\sim DP(\alpha, G_0) \\ \forall i &\in \mathbb{N} \end{aligned} \tag{8}$$

The Blackwell-MacQueen urn scheme refers to a sequence of predictive distributions for $\{\theta_i\}_i$, using the notation of (8). [7] proved that the distribution given by (9) converges (with respect to n) almost surely to a discrete distribution, G, where G is a realisation of a Dirichlet Process. Before looking at these predictive distributions, the following properties are required.

Theorem
Properties of Dirichlet Process

Let $G \sim DP(\alpha, G_0)$, and $\theta_1, \ldots, \theta_n \in A$ are random variables distributed according to G.

1. Then the posterior:

$$G|\{\theta_1, \ldots, \theta_n\} \sim DP(\alpha + n, \frac{\sum_{k=1}^{n} \delta_{\theta_i}(\cdot) + \alpha G_0(\cdot)}{\alpha + n})$$

2. The expectation of the Dirichlet Process is its base distribution:

$$\mathbb{E}[G] = G_0$$

Or more formally, for any $B \subseteq A$, $\mathbb{E}[G(B)] = G_0(B)$

Proof. □

1. Proofs given in [7, 5, 2, 1] .
2. Using Ferguson's definition:

$$(G(B), G(B^C)) \sim \text{Dirichlet}(\alpha G_0(B), \alpha G_0(B^C))$$

In addition, from (1):

$$\mathbb{E}[P_1^{r_1} P_{2,}^{r_2} \ldots, P_K^{r_K}] = \frac{\Gamma(\sum_{i=1}^{K} \alpha_i)}{\Gamma(\sum_{i=1}^{K} \alpha_i + R)} \frac{\prod_{i=1}^{K} \Gamma(\alpha_i + r_i)}{\prod_{i=1}^{K} \Gamma(\alpha_i)}$$

Therefore, using $P_1 = G(B)$, $P_2 = G(B^C)$, $\alpha_1 = \alpha G_0(B)$, $\alpha_2 = \alpha G_0(B^C)$:

$$\mathbb{E}[G(B)] = \frac{\Gamma(\alpha(G_0(B) + G_0(B^c)))}{\Gamma(\alpha(G_0(B) + G_0(B^c)) + 1)} \frac{\Gamma(\alpha G_0(B) + 1)}{\Gamma(\alpha G_0(B))} \frac{\Gamma(\alpha G_0(B^c))}{\Gamma(\alpha G_0(B^c))}$$

As $\Gamma(\alpha + 1) = \alpha\Gamma(\alpha)$, and $G_0(B) + G_0(B^c)$=1:

$$\begin{aligned} \mathbb{E}[G(B)] &= \frac{\Gamma(\alpha)}{\Gamma(\alpha + 1)} \frac{\Gamma(\alpha G_0(B) + 1)}{\Gamma(\alpha G_0(B))} \\ &= \frac{1}{\alpha} \frac{\alpha G_0(B)}{1} \\ &= G_0(B) \end{aligned}$$

This urn scheme is used to generate draws from a Dirichlet Process G, without explicitly constructing G. This is done by marginalising out G in the posterior distribution. However, it is important to notice that the draws from this urn scheme, $\{\theta_i\}_i$, will no longer be independent once G is marginalised out. The Blackwell-MacQueen urn scheme has been used to show the the existence of the Dirichlet Process and that the predictive distribution of θ_n conditioned on $\theta_{1:(n-1)}$ is the same as the base distribution of the posterior Dirichlet Process. The predictive distribution of θ_n conditioned on $\theta_{1:(n-1)}$ can be summarised in the next corollary.

Corollary
Urn Scheme:

$$\theta_n|\{\theta_1 = x_1, \ldots, \theta_{n-1} = x_{n-1}\} \sim \frac{\sum_{i=1}^{n-1} \delta_{x_i}(\cdot)}{\alpha + n - 1} + \frac{\alpha G_0(\cdot)}{\alpha + n - 1} \tag{9}$$

where $\theta_i|G \overset{iid}{\sim} G$, $\forall i$ and $G \sim DP(\alpha, G_0)$

Proof. For measurable $B \in \mathcal{A}$, the predictive distribution is defined by marginalising out G, which is essentially "integrating out the infinite part" [10]:

$$\begin{aligned}
\mathbb{P}[\theta_n \in B|x_1, \ldots x_{n-1}] &= \mathbb{E}[\mathbb{I}_B(\theta_n)|x_1, \ldots x_{n-1}] \\
&= \mathbb{E}_G[\mathbb{E}[\mathbb{I}_B(\theta_n)|x_1, \ldots x_{n-1}, G]|x_1, \ldots x_{n-1}] \\
&= \mathbb{E}_G[G(B)|x_1, \ldots x_{n-1}] = \frac{\sum_{i=1}^{n-1} \delta_{x_i}(B)}{\alpha + n - 1} + \frac{\alpha G_0(B)}{\alpha + n - 1}
\end{aligned}$$

The second step uses the Tower Property of expectation and the third uses the independence of the $\{\theta_i\}_i$ when G is known. The last step is justified as the expectation of the posterior Dirichlet Process distributed random measure evaluated at B is the posterior base distribution evaluated at B. This last step is marginalising out the posterior of G. □

The convenient form of this predictive distribution is useful in expressing the distribution of $\theta_1, \ldots, \theta_k|\alpha, G_0$ assuming the structure given by (8). In shorthand, the probability density corresponding to $\{\theta_i\}_i$ is:

$$P(\theta_1, \ldots, \theta_n) = P(\theta_1) \prod_{i=2}^{n} P(\theta_i|\theta_1, \ldots, \theta_{i-1})$$

where the first $\theta_1 \sim G_0$ and the conditional distributions $P(\theta_i|\theta_1, \ldots, \theta_{i-1})$ are given by (9)

The concept of exchangeability gives justification to a Bayesian approach. A more complete discussion is given by [9], however, a brief summary is given here.

Definition
Infinitely Exchangeable

A sequence of random variables $\{\theta_i|i \geq 1\}$ is infinitely exchangeable if $\forall n \in \mathbb{N}$:
$(\theta_1, \ldots, \theta_n)$ has the same distribution as $(\theta_{\sigma(1)}, \ldots, \theta_{\sigma(n)})$ for any permutation σ of $\{1, \ldots, n\}$

The density given by (12) shows that the parameters $\theta_{1:n}$ are exchangeable in the urn scheme. This is important for sampling from the conditional distributions as one can treat sampling from $\theta_i|\theta_{-i}$ as sampling the n^{th} parameter in a permutation, and hence the predictive distribution given by the urn scheme may be used.

The Blackwell-MacQueen urn scheme name is derived from an analogy used to interpret the predictive distribution. The analogy is as follows: let θ_i represent the colour of the i^{th} ball drawn from an urn, A is the set of colours of balls. θ_n is generated according to the steps below:

1. Start with an empty urn

2. Draw a colour, θ_1, from the non-atomic distribution G_0, add a ball of that colour to the urn

3. For $i \in \{2, \dots n\}$, let θ_i be the colour of the i^{th} ball

   (a) With probability $\frac{i-1}{\alpha+i-1}$ pick a ball uniformly from the urn, add the drawn ball and another ball of the same colour (θ_i) to the urn

   (b) With probability $\frac{\alpha}{\alpha+i-1}$ draw a colour θ_i from the distribution G_0, add a ball of colour θ_i to the urn

This generates the predictive distribution given by (9).

2.5 Chinese Restaurant Process

The Blackwell-MacQueen Urn scheme illustrates the discreteness property of the Dirichlet Process, meaning that there is a positive probability of repeated draws from G (where$G \sim \mathrm{DP}(\alpha, G_0)$), regardless of the continuity of the base distribution G_0. Let $\theta_{1,}\theta_2, \dots \theta_{n-1}$ be draws from G. As draws have positive probability of being repeated denote $\phi^*_{1,}\phi^*_2, \dots, \phi^*_m$ as the unique values, and let n_k be the number of θ_i, $i \in \{1, \dots n-1\}$ such that $\theta_i = \phi^*_k$. The allocation variables $\{Z_i\}_{i=1}^n$may be introduced to express the partition and simplifies the notation as follows: $\theta_i = \phi^*_{z_i}$. Under this labelling, the predictive distribution, (9), can be written as:

$$\theta_n | \{\theta_1 = x_1, \dots, \theta_{n-1} = x_{n-1}\} \sim \frac{\sum_{k=1}^m n_k \delta_{\phi^*_k}}{\alpha + n - 1} + \frac{\alpha G_0}{\alpha + n - 1} \tag{10}$$

Or equivalently, where $\{Z_i\}_{i=1}^{n-1}$ takes vales in $\{1, \dots, m\}$

$$Z_n | \{Z_1 = z_1, \dots, Z_{n-1} = z_{n-1}\} \sim \frac{\sum_{k=1}^m n_k \delta_k}{\alpha + n - 1} + \frac{\alpha \delta_{m+1}}{\alpha + n - 1} \tag{11}$$

$$\phi^*_k \overset{iid}{\sim} G_0$$

The allocation values $\{Z_i\}_{i=1}^{n-1}$ partition the $n-1$ draws, $\theta_{1,}\theta_2, \dots \theta_{n-1}$, into m clusters, whereby the draw θ_i is allocated to cluster k if $Z_i = k$. The terms "cluster" and "component" shall be used interchangeably. It is clear from (10) that the larger the k^{th} cluster, the greater chance that the next draw will be allocated to cluster k, this is known as the "rich get richer" property. As the $\theta_{1,}\theta_2, \dots, \theta_{n-1}$ are random variables, the partitioning is itself random and is said to be distributed according to a Chinese Restaurant Process (CRP), this partitioning is fully described by, $\{Z_i\}_{i=1}^N$. Therefore, the variables $\theta_{1,}\theta_2, \dots, \theta_n$ can be fully described by $\phi^*_{1,}\phi^*_2, \dots, \phi^*_m$ and $\{Z_i\}_{i=1}^n$.

The CRP can be used to generate draws from $G \sim \mathrm{DP}(\alpha, G_0)$:

1. Generate a partition of $\{1, \dots n\}$ according to the CRP, denote the number of clusters as m and let z_i be the cluster allocation of i

2. For each of the m clusters generated by step 1, draw a value from the base distribution, $\phi^*_j \sim G_0$

3. $\forall i$ such that $z_i = j$, set $\theta_i = \phi^*_j$, $j \in \{1, \dots, m\}$.

The Chinese Restaurant Process name comes from yet another analogy, similar to the urn scheme. In this analogy there are an infinite number of tables in a Chinese restaurant, with n_k customers at table $k \in \{1, \dots, m\}$. Only m of the tables are occupied. A new customer enters the restaurant. With probability proportional to n_k the customer will sit at table k, with probability proportional to α the customer sits at an empty table. This is almost identical to the urn analogy except there is no draw from the base distribution associated with each table.

Corollary

The probability density for this CRP is the same as that for the urn scheme. Using the same notation as above for $z_{1:n}$, $\phi^*_{1:m}$ and , where $\theta_i = \phi^*_{z_i}$:

$$P(z_1, \ldots, z_n, \phi^*_1, \ldots, \phi^*_m) \quad = \quad P(\theta_1, \ldots, \theta_n)$$

Proof. Recall from the urn scheme:

$$P(\theta_1, \ldots, \theta_n) = P(\theta_1) \prod_{i=2}^{n} P(\theta_i | \theta_1, \ldots, \theta_{i-1})$$

Let g_0 be the density corresponding to the base distribution G_0, from the urn representation:

$$\begin{aligned} P(\theta_1, \ldots, \theta_n) &= g_0(\theta_1) \prod_{i=2}^{n} \left(\frac{\sum_{j=1}^{i-1} \mathbb{I}_{\{\theta_j\}}(\theta_i)}{\alpha + i - 1} + \frac{\alpha g_0(\theta_i) \mathbb{I}_{\{\theta_{new}\}}(\theta_i)}{\alpha + i - 1} \right) \\ &= g_0(\phi^*_{z_1}) \prod_{i=2}^{n} \left(\frac{\sum_{j=1}^{m(i)} n_j(i) \mathbb{I}_{\{\phi^*_j\}}(\phi^*_{z_i})}{\alpha + i - 1} + \frac{\alpha g_0(\theta_i) \mathbb{I}_{\{\phi^* new\}}(\phi^*_{z_i})}{\alpha + i - 1} \right) \end{aligned}$$

Let $n_j(i)$ be the number of $\theta_{1:(i-1)}$ equal to ϕ^*_j and $m(i)$ be the number of unique $\theta_{1:(i-1)}$. The $n_j(i)$ and $m(i)$ are dependent on the ordering $\theta_1, \ldots, \theta_n$. Let n_j be the number of $\theta_1, \ldots, \theta_n$ which are equal to ϕ^*_j. $n_j(1) = 0$ and increase by 1 or 0 in of product terms to reach n_j. In addition, the unique $\{\phi^*_{z_i}\}_{i=1}^n$ are fixed with count m. Note, m is a function of $Z_1, \ldots, Z_n$. The denominator $(\alpha + i - 1)$ increases by 1 as i rises. Hence by expansion and simplification:

$$\begin{aligned} P(\theta_1, \ldots, \theta_n) &= \frac{\prod_{k=1}^{m} \alpha g_0(\phi^*_k) n_k!}{(\alpha + n - 1)!} && (12) \\ &= \prod_{k=1}^{m} g_0(\phi^*_k) \frac{\prod_{k=1}^{m} \alpha n_k!}{(\alpha + n - 1)!} && (13) \\ &= P(\phi^*_1, \ldots, \phi^*_m | m) P(z_1, \ldots, z_n) && (14) \\ &= P(z_1, \ldots, z_n, \phi^*_1, \ldots, \phi^*_m) && (15) \end{aligned}$$

Where:

$$P(z_1, \ldots, z_n) = P(z_1) \prod_{i=2}^{n} P(z_i | z_1, \ldots, z_{i-1})$$

and $P(z_i | z_1, \ldots, z_{i-1})$ is given by equation (11). By a similar argument to above (without the $g_0(\cdot)$ terms):

$$\begin{aligned} P(z_1, \ldots, z_n) &= \frac{\prod_{k=1}^{m} \alpha n_k!}{(\alpha + n - 1)!} \\ P(\phi^*_1, \ldots, \phi^*_m | z_1, \ldots, z_n) &= \prod_{k=1}^{m} g_0(\phi^*_k) \end{aligned}$$

□

Since equation (12) does not depend on the order of the $\{Z_i\}_i$ or $\{\theta_i\}_i$, the exhangeability condition holds. This shall be used later in this report.

2.6 Parameter Interpretation

The various representations of the Dirichlet Process illustrate how the choice of parameters, α and G_0, influence the form of the Dirichlet Process G. Informally, the stick-breaking construction, $G = \sum_{k=1}^{\infty} \pi_k \delta_{\phi_k^*}$, shows that the Dirichlet Process G is a discretization of the base distribution. By this, it is meant that G is a discrete distribution with the same support as the base distribution, and "on average" (the expected value of) the random distribution G will yield the same value as that of its base distribution i.e. $\mathbb{E}[G(B)] = G_0(B)$.

Without requiring detailed knowledge of the Beta distribution, the urn representation and the Chinese restaurant process show how the concentration parameter, α, influences the probability masses of the Dirichlet Process, G. The higher the value of α, the more likely it is for a "new colour" to be drawn from the urn in the urn scheme analogy, or the more probable a "customer" sits at a "new table" in the Chinese restaurant process representation. In terms of the distribution G, the larger the value of α, the fewer large probability masses, $\{\pi_k\}_k$, there will be. Again using the Stick Breaking Construction, the probability masses are, $\{\pi_k\}_k$:

$$\beta_k \overset{iid}{\sim} \text{Beta}(1, \alpha),\ \pi_k = \beta_k \prod_{j=1}^{k-1} (1 - \beta_j)$$

The expected value of β_k proportion is $\frac{1}{\alpha}$, therefore the larger the α, the smaller the expected proportion of the "stick broken off" at each iteration.

3 Bayesian Inference on Mixture Models

Mixture models are a flexible class of models which can be used to model data associated with sub-populations, often referred to as clusters or components. Similar to the previous sections, let A be the set of possible parameters values, with corresponding Borel σ-algebra, $\mathcal{A}$. This space, $(A, \mathcal{A})$, will be a parameter space. Denote a specified parametric family of distributions by $\mathcal{F} = \{F(\cdot, \theta)|\theta \in A\}$, where $F(\cdot, \theta)$ is a specified distribution with parameter θ. Let Y_i, $i \in \mathbb{N}$ be random variables representing observable data taking values in set $\mathcal{Y}$, with corresponding Borel σ-algebra $\mathcal{B}$. The parameters θ are treated as non-observable, latent, random variables. Informally, a mixture model is a distribution on $(\mathcal{Y}, \mathcal{B})$ represented by a combination of distributions $F(\cdot, \theta)$.

3.1 Finite Mixture Model

$$Y_i|\boldsymbol{\pi}, \boldsymbol{\phi^*} \overset{iid}{\sim} \sum_{k=1}^{m} \pi_k F(\cdot, \phi_k^*)$$

$\boldsymbol{\phi^*} = \phi_{1:m}^*$, $\boldsymbol{\pi} = \pi_{1:m}$
In a finite mixture model, it is assumed that the observed data, $\{y_i,\ i \in \{1, \ldots, N\}\}$, are realisation of i.i.d random variables, Y_i, distributed according to a mixture of a finite number of component distributions, $F(\cdot, \phi_k^*)$ $\phi_k^* \in A$, $k \in \{1, \ldots, m\}$ for some positive integer m. The mixing is performed by summing the distributions with weights, known as mixing probabilities, given by $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_m)$. It is often convenient to introduce allocation variables to simplify the mixture model. For each of these distributions let $Z_i \in \{1, \ldots, m\}$ be a latent variable representing the component label for the i^{th} data point, m is the number of components. $\mathbb{P}[Z_i = k] = \pi_k$, $k \in \{1, \ldots, m\}$ and $Y_i|z_i, \phi_{z_i}^* \sim F(\cdot, \phi_{z_i}^*)$. The representations with and without the latent variables are describe the same process in which generates the data $\{y_i\}_i$.

The mixture model is a generative model in the sense that it prescribes a way in which data can be generated. A draw from the mixture distribution is carried out by first choosing a mixture component, labelled z_i, with probability given by π_{z_i}, then the observable data-point is drawn from the corresponding distribution associated with the chosen component, $F(\cdot, \phi_{z_i}^*)$. Each component has an associated parameter, ϕ_k^*, $k \in \{1, 2 \ldots, m\}$. For general mixture models, the component distributions,F_{z_i}, associated with each component, z_i, can be any distribution. However, for convenience let the component distributions be from the same parametric family.

A Bayesian approach to mixture models is to treat parameters as random with their own distributions, known as priors. The random parameters of interest are mixing probabilities, $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_m)$, and component parameters $\{\phi_k^*\}_{k=1}^m$. The standard priors are the Dirichlet distribution prior for $\boldsymbol{\pi}$, and a base distribution, G_0, for the component parameters. The component parameter associated to each data-point i is denoted θ_i, which is equal to $\phi_{z_i}^*$.
A full summary is given by the following table:

Representation	1	2
Dirichlet Specific Part	$\boldsymbol{\pi}\|\boldsymbol{\alpha} \sim \text{Dirichlet}(\boldsymbol{\alpha})$	$\boldsymbol{\pi}\|\boldsymbol{\alpha} \sim \text{Dirichlet}(\boldsymbol{\alpha})$
General Mixture Model Part	$\phi_k^*\|G_0 \overset{iid}{\sim} G_0$ $Y_i\|\boldsymbol{\pi}, \boldsymbol{\phi^*} \overset{iid}{\sim} \sum_{k=1}^m \pi_k F(\cdot, \phi_k^*)$	$z_i\|\boldsymbol{\pi} \sim \text{Categorical}(\boldsymbol{\pi})$ $\phi_k^*\|G_0 \overset{iid}{\sim} G_0$ $Y_i\|z_i, \phi_{z_i}^* \overset{iid}{\sim} F(\cdot, \phi_{z_i}^*)$

Table 1: Finite Dirichlet Mixture Model

It is straightforward to show equivalence:

In representation 1:

$$\mathbb{P}[Y_i \in B|\boldsymbol{\pi}, \boldsymbol{\phi^*}] = \sum_{k=1}^{m} \pi_k F(B, \phi_k^*)$$

In representation 2:

$$\begin{aligned}\mathbb{P}[Y \in B|\boldsymbol{\pi}, \boldsymbol{\phi^*}] &= \mathbb{E}_{Z_i}[\mathbb{P}([Y \in B|Z_i = z_i, \boldsymbol{\pi}, \boldsymbol{\phi^*}]|\boldsymbol{\pi}, \boldsymbol{\phi^*}] \\ &= \mathbb{E}_{Z_i}[F(B, \phi_{z_i}^*)|\boldsymbol{\pi}, \boldsymbol{\phi^*}] \\ &= \sum_{k=1}^{m} \pi_k F(B, \phi_k^*)\end{aligned}$$

3.2 Dirichlet Process Mixture Model

In this Bayesian framework, where parameters are treated as random, the DP Mixture Model (DPMM) is defined by the following distribution on measurable space $(\mathcal{Y}, \mathcal{B})$, which is a concise expression given by [10]:

$$\mathrm{M}(\cdot, G) = \int F(\cdot, \theta) dG(\theta) = \sum_{i=1}^{\infty} \pi_i F(\cdot, \phi_i^*)$$

where G is some Dirichlet Process, F is some distribution from a given parametric family.

The DPMM can be viewed as a generative model:

Representation:	1	2
Dirichlet Process Specific Part	$G \sim \mathrm{DP}(\alpha, G_0)$	$\boldsymbol{\pi}\|\boldsymbol{\alpha} \sim \mathrm{GEM}(\boldsymbol{\alpha})$
General Mixture Model Part	$\theta_k\|G \sim G$ $y_i\|\theta_i \overset{iid}{\sim} F(\cdot, \theta_i), \forall i$	$z_i\|\boldsymbol{\pi} \sim \mathrm{Categorical}(\boldsymbol{\pi})$ $\phi_k^*\|G_0 \sim G_0$ $y_i\|z_i, \{\phi_k^*\}_k \overset{iid}{\sim} F(\cdot, \phi_{z_i}^*), \forall i$

Table 2: Dirichlet Process Mixture Model

Following representation 1, the data is generated from the mixture model as follows: a distribution G is a realised Dirichlet Process, latent parameters θ_i $i \in \mathbb{N}$ are drawn from G, then the observable data, Y_i are drawn from $F(\cdot, \theta_i)$. Representation 2 is equivalent using the Stick Breaking construction, along with additional latent variables Z_i which specify component allocations and ϕ_k^* which are the unique parameters, referred to as component parameters. A probability vector $\boldsymbol{\pi}$ is drawn from the specified GEM($\boldsymbol{\alpha}$) distribution, a component allocation is drawn from the Categorical($\boldsymbol{\pi}$) distribution, component parameter $\phi_{z_i}^*$ is drawn from specified base distribution G_0. Finally the data point,Y_i, is drawn from $F(\cdot, \phi_{z_i}^*)$.

Both representations are given in the plate diagrams below, which also illustrate the dependency structure of the variables and parameters. The variables in the non-highlighted box are the observed variables (the data) and those in the coloured boxes are latent. The arrows illustrate how each variable depends on another. For example, the $\{\theta_i\}_i$ are conditionally independent given G, but dependent otherwise (this is how exchangeability arises [9]). Each θ_i is conditionally independent of both α and G_0 given G. The boxes around variables indicates repetitions of the variables which have the same dependencies.

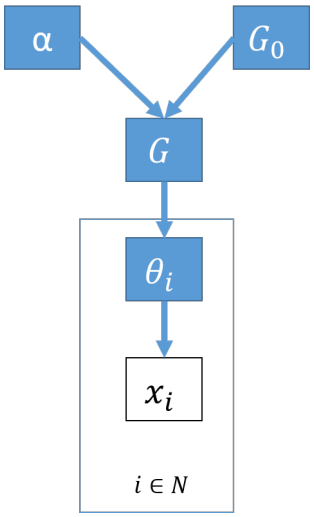


Figure 1: Plate Diagram: Representation 1

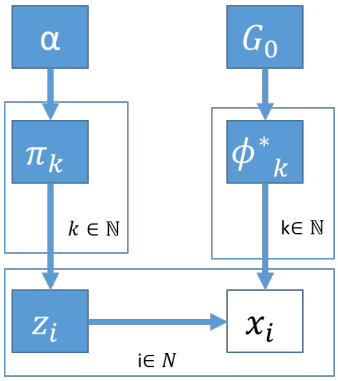


Figure 2: Plate Diagram: Representation 2

3.3 Bayesian Inference

It is assumed throughout that the data being modelled was generated according to some Dirichlet Process mixture model with unknown parameters. Mixture models describe a probabilistic mechanism to generate data. The challenge now is to perform inference, meaning gaining insights into the hidden parameters, given a set of data. Informally, this means learning what parameters are "likely" to have generated the data. The posterior density of the parameters is: $P(\theta_{1:n}|y_{1:n})$ or $P(\phi^*_{1:m}, Z_{1:n}|y_{1:n})$ depending on the representation. Through Bayes' theorem, it can be seen that the distributions of interest for each representation are:
Representation 1 (Urn Representation):

$$P(\theta_{1:n}|y_{1:n}) \propto P(y_{1:n}|\theta_{1:n})P(\theta_{1:n}) \propto \left(\prod_{i=1}^{n} f(y_i|\theta_i)\right) P(\theta_{1:n})$$

Representation 2 (CRP Representation):

$$P(\phi^*_{1:m}, z_{1:n}|y_{1:n}) \propto \left(\prod_{i=1}^{n} f(y_i|\phi^*_{z_i})\right)\left(\prod_{k=1}^{m} g_0(\phi^*_k)\right) P(z_{1:n})$$

The aim of Bayesian inference is to compute integrals (equivalently expected values) with respect to these, i.e. for an integrable function h:

$$\mathbb{E}_{P(\theta_{1:n}|y_{1:n})}[h(\theta_{i:n})]$$

Markov Chain Monte Carlo methods provide a solution when the distribution $P(\theta_{1:n}|y_{1:n})$ is difficult to work with analytically.

4 Markov Chain Monte Carlo

When it is difficult or not possible to analytically evaluate expected values, Monte Carlo techniques can be used to calculate numerical approximations. Markov Chain Monte Carlo (MCMC) techniques are used to obtain an *approximate* sample from a distribution of interest, referred to as a target distribution, without sampling from the distribution directly. MCMC involves simulating a Markov Chain whose invariant distribution is the target distribution. Simulating a Markov Chain means recording numerical realisations of the states in the chain. If $\{x_i\}_{i\in\mathbb{N}}$ are the values of the states drawn in a MCMC procedure, with Markov chain $\{X_i\}_i$, where the limiting distribution is F then I shall use informal terminology and call the MCMC procedure *valid* when, for any integrable function g:

$$\frac{1}{n}\sum_{i=1}^{n} g(X_i) \xrightarrow[n\to\infty]{} \mathbb{E}_F[g(X)] \text{ a.s.} \tag{16}$$

where $X \sim F$. More formally, the Markov Chain or its associated kernel is ergodic.

In an attempt to be self-contained, a brief summary of the main MCMC concepts are detailed below. A more complete and detailed description, including relevant proofs, of the concepts below is given by [11], the notation below has been adopted from the same paper .

4.1 Markov Chains

For the purpose of this work, we shall consider Markov Chains as time homogeneous stochastic processes with an ordered index set that satisfy the Markov property, in particular the index set of natural numbers $\mathbb{N}$ shall be used. The value of the random variable at each index entry is known as the state of the Markov chain. The Markov property can intuitively be thought of by the idea that by knowing the current state, the distribution of the next state is independent of all previous states. The transition probability kernel is a function $\mathcal{K} :\ E \times \mathcal{B}(E) \to \mathbb{R}$,

$$\mathcal{K}(x, B) = \mathbb{P}[X' \in B | X = x]$$

where $\mathcal{K}(\cdot, B)$ is a measurable function for fixed $B \subseteq E$, and $\mathcal{K}(x, \cdot)$ is a probability distribution $\forall x \in E$.

Definition
Reversibility and Detailed Balance Equations
Let $\{X_n\}_{n\in\mathbb{N}}$ be a Markov Chain on state space E, with probability law $\mathcal{K}(x, B) = \mathbb{P}[X' \in B | X = x]$. $\{X_n\}_{n\in\mathbb{N}}$ is reversible with respect to probability distribution $\pi(\cdot)$ if it satisfies the detailed balance equations, $\forall x, y \in E$:

$$\pi(dx)\mathcal{K}(x, dy) = \pi(dy)\mathcal{K}(y, dx)$$

Definition
Invariant Distribution
A Markov Chain $\{X_n\}_{n\in\mathbb{N}}$ with state space E has invariant distribution $\pi(\cdot)$ if:

$$\int_B \pi(dy) = \int_B \int_E \pi(dx)\mathcal{K}(x, dy)$$

$\forall B \in \mathcal{B}(E)$, where $\mathcal{K}$ is the transition probability kernel

Corollary
A Markov Chain $\{X_n\}_{n\in\mathbb{N}}$ with state space E which is reversible with respect to distribution $\pi(\cdot)$, also has invariant distribution $\pi(\cdot)$, $\forall B \in \mathcal{B}(E)$.

Proof. $\int_{x\in A}\pi(dx)\mathcal{K}(x,dy)=\int_{x\in A}\pi(dy)\mathcal{K}(y,dx)=\pi(dy)\int_{x\in A}\mathcal{K}(y,dx)=\pi(dy)$ □

Definition
Aperiodic
A Markov Chain with stationary distribution $\pi(\cdot)$ is aperiodic if there do not exist disjoint sets $A_1,\ldots,A_d\subset E$, $d\geq 2$, where $\mathcal{K}(x,A_{i+1})=1$, for all $x\in A_i$ $i\in\{1,\ldots,d-1\}$and $\mathcal{K}(x,A_1)=1$ for $x\in A_d$ such that $\pi(A_i)>0\ \forall i$

Definition
$\mu-$irreducible
Let $\{X_n\}_{n\in\mathbb{N}}$ be a Markov Chain on state space E, with probability law $\mathcal{K}(x,B)=\mathbb{P}[X'\in B|X=x]$. $\{X_n\}_{n\in\mathbb{N}}$ is $\mu-$irreducible if there exists a non-zero $\sigma-$finite measure μ such that $\forall A\in\mathcal{B}(E)$ where $\mu(A)>0$, $\forall x\in E\ \exists n\in\mathbb{N}$ such that $\mathcal{K}^n(x,A)>0$. If $\forall A\in\mathcal{B}(E)$, $n=1$ in the above definition, then the chain is called strongly irreducible, and this implies aperiodicity.

Theorem
[11] If a Markov Chain on a state space E, with countable generated $\sigma-$algebra, is $\mu-$irreducible and aperiodic, and has a stationary distribution $\pi(\cdot)$, then for π-almost all $x\in E$:

$$\lim_{n\to\infty}\|\mathcal{K}^n(x,\cdot)-\pi(\cdot)\|=0$$

Under the same conditions, for any integrable function $\varphi:E\to\mathbb{R}$:

$$\lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n}\varphi(x_i)\to\int\varphi(x)d\pi(x)\text{ with probability 1}$$

This is a very general theorem with relatively weak conditions, a thorough proof given in [11]. Most MCMC procedures are designed to have the correct invariant distribution (as shall be shown for two specific cases in the next section). In addition, the measure μ need not be the invariant distribution but simply the Lebesgue measure for example. The Borel σ-algebra on the reals is countably generated by subsets of the rationals, therefore this condition is often not an issue. The remaining conditions are irreducibility and aperiodicity, which shall be explained for the presented algorithms.

4.2 Metropolis Hastings and Gibbs Sampling

The Metropolis Hastings (MH) and Gibbs sampling algorithms are set procedures used to perform MCMC. The algorithms guarantee the correct invariant distribution, however do not guarantee that the chains are ergodic, which is determined by the case specific target distribution and proposals.

Metropolis Hasting Algorithm

Start with some initial value $X^{(0)}=(X_1^{(0)},\ldots,X_p^{(0)})$ for some $p-$dimensional state of Markov Chain $\{X^{(n)}\}_{n\in\mathbb{N}}$.For $t\in\{1,\ldots,n\}$:

1. Draw $X\sim q(\cdot|X^{(t-1)})$
2. Set $X^{(t)}=X$ with probability $\alpha(X^{(t-1)},X)=\min\left\{1,\frac{f(X)q(X^{(t-1)}|X)}{f(X^{(t-1)})q(X|X^{(t-1)})}\right\}$, otherwise set $X^{(t)}=X^{(t-1)}$

In the MH algorithm f is the density of the target distribution, $q(\cdot|X^{(t-1)})$ is the proposal distribution.
The transition kernel density for the MH algorithm is given by:

$$K(x^{(t-1)},x^{(t)})=\alpha(x^{(t-1)},x^{(t)})q(x^{(t)},|x)+(1-\alpha(x^{(t-1)})\delta_{x^{(t-1)}}(x^{(t)})$$

where $\alpha(x^{(t-1)}) = \int \alpha(x, x^{(t-1)}) q(x|x^{(t-1)}) dx$ is the probability that a newly proposed state is accepted.

It may be challenging to sample from a multivariate distribution using Metropolis Hastings for multiple variables at each iteration. Often it is simpler to sample one dimension at a time, and update the state after MH updates for each variable in the state. A specific example of this is Gibbs sampling.

Gibbs Sampling

Start with some initial value $X^{(0)} = (X_1^{(0)}, \ldots, X_p^{(0)})$ for some $p-$dimensional state of Markov Chain $\{X^{(n)}\}_{n \in \mathbb{N}}$.

1. For $t \in \{1, \ldots, n\}$:
   (a) For $i \in \{1, \ldots, p\}$:
      i. Draw $X_i \sim f_{X_i|X_{-i}}(\cdot|x_{1:(i-1)}^{(t)}, x_{(i+1):p}^{(t-1)})$
      ii. Set $X_i^{(t)} = X_i$

This has the transition kernel with density:

$$K(x^{(t)}, x^{(t+1)}) = \prod_{i=1}^{p} f_{X_i|X_{-i}}(x_i^{(t)})$$

Gibbs sampling is an example of a cycle of kernels, one for updating each element of the state. If it is difficult to draw from the distribution with density $f_{X_i|X_{-i}}$ then it is possible to use a Metropolis-Hastings step to take this sample within the Gibbs sampling framework for the complete state. This shall be referred to as "MH within Gibbs".

4.3 Auxiliary Variables and Cycles of Kernels

Auxiliary Variables

It is possible to take a MCMC sample from a target distribution π_X of random variable X by first sampling from a target distribution $\pi_{X,Y}$ of X, Y, which has marginal distribution π_X once the auxiliary variables, Y, are marginalised. This is a useful procedure to sample from π_X if it is more convenient to sample from $\pi_{X,Y}$. Consider the distribution with density $\pi_X(x)$ and the joint distribution $\pi_{X,Y}(x, y) = \pi_X(x)\pi_{Y|X}(y)$. If it is possible to simulate a Markov Chain on the extended state (X, Y) with kernel $K((x, y); (x', y'))$ and invariant distribution $\pi_{X,Y}$, then one can extract a Markov Chain with invariant distribution π_X by ignoring the auxiliary variables in the state providing $\int \pi_{X,Y}(x, y)dy = \pi_X(x)$.

In a similar fashion, if it is possible to simulate a Markov Chain for $\pi_{X,Y}$, the kernel may be modified to provide a kernel for state X with invariant distribution π_X where again $\pi_{X,Y}(x, y) = \pi_X(x)\pi_{Y|X}(y)$. The motivation for this can be seen in algorithm 3 of this dissertation for clustering data-points using the Dirichlet Process Mixture Model component allocations,where the cluster parameters can be marginalised out. It is assumed a kernel $K((x, y); (x', y'))$ is known such that:

$$\int \int K((x, y); (x', y')) \pi_{X,Y}(x, y)dxdy = \pi_{X,Y}(x', y')$$

The aim is to find K_X such that:

$$\int K_X(x, x') \pi_X(x)dx = \pi_X(x')$$

$$\begin{aligned}\pi_X(x') &= \int \pi_{X,Y}(x',y')dy \\ &= \int_{y'}\int_x\int_y K\left((x,y);(x',y')\right)\pi_{X,Y}(x,y)dxdydy' \\ &= \int_{y'}\int_x\int_y K\left((x,y);(x',y')\right)\pi_X(x)\pi_{Y|X}(y)dxdydy'\end{aligned}$$

The order of integration may be swapped using Fubini-Tonelli given that the terms inside the integrals are positive.

$$\begin{aligned}\pi_X(x') &= \int_{y'}\int_x\int_y K\left((x,y);(x',y')\right)\pi_X(x)\pi_{Y|X}(y)dxdydy' \\ &= \int_x \pi_X(x)\left(\int\int K\left((x,y);(x',y')\right)\pi_{Y|X}(y)dydy'\right)dx\end{aligned}$$

Hence:

$$K_X\left(x,x'\right) = \int\int K\left((x,y);(x',y')\right)\pi_{Y|X}(y)dydy'$$

Cycles of Kernels

Cycles of kernels will be used in all the algorithms presented. Consider the state $X_{1:p} = (X_1, X_2, \ldots, X_p)$ for some natural number p. Each of the kernels in the cycle may update the whole state, $X_{1:p}$, or as can be done with Gibbs sampling, may only partially update the state. If each kernel has the desired invariant distribution, then the cycle has the invariant distribution. It is sufficient to show this for a cycle of two kernels with densities $K_1$and K_2, then appeal to induction. Consider the following cycle of kernels: $K(x,y) = \int K_1(x,x')K_2(x',y)dx'$.

$$\begin{aligned}\int \pi_X(x)\left(\int K_1(x,x')K_2(x',y)dx'\right)dx &= \left(\int \pi_X(x)\int K_2(x,x')dx\right)K_1(x',y)dx' \\ &= \int \pi_X(x')K_2(x',y)dx' \\ &= \pi_X(y)\end{aligned}$$

A cycle of kernels, $\{K^{(i)}\}_{i=1}^R$, may update the whole state, in which case the combined kernel would be, K:

$$K(x,x') = \int\ldots\int K^{(1)}(x,x^{(1)})\left(\prod_{i=2}^{R-1}K^{(i)}(x^{(i-1)},x^{(i)})\right)K^{(R)}(x^{(R-1)},x')dx^{(1)}\ldots dx^{(R-1)}$$

5 Algorithms

The algorithms presented in [3] will be investigated, in particular, those numbered 1, 2, 3, 5, 6, 7, 8. Algorithm 4[34] has been omitted due to its similarity to algorithm 8. These algorithms can be grouped into conjugate algorithms (1,2,3) and non-conjugate algorithms (5, 6, 7, 8). The algorithms are labelled conjugate as they require the choice that G_0 is a conjugate prior for the parameter θ, with likelihood $f(x, \theta)$, for some data-point x. This simply means the distribution of θ given x is of the same parametric family as G_0, hence the quantity $\int f(x, \theta) dG_0(\theta)$ may be calculated analytically. For the non-conjugate algorithms it is not required that one is able to calculate the integral $\int f(x, \theta) dG_0(\theta)$ or sample directly from the posterior distribution of θ given $x_{1:n}$.

It can also be seen that algorithms 1 and 6 use the Blackwell-MacQueen urn scheme for posterior sampling, where algorithm 6 uses a Metropolis Hastings step with the conditional prior for parameters $\{\theta_i\}_i$ as proposal distribution to circumnavigate the non-conjugacy issues. Similarly, algorithms 2 and 5 use the Chinese Restaurant Process representation, where again algorithm 5 includes a Metropolis Hastings step using the conditional prior for component allocations as the proposal. Algorithm 3 is similar to algorithm 2, but with the component parameters, $\phi^*_{1:m}$, marginalised out of the state in the Markov Chain. Just as algorithm 3 uses a reduced state in the Markov Chain, algorithm 8 uses auxiliary variables to first temporarily extend the state, then these auxiliary variables are marginalised. It can be seen that this allows for one to avoid conjugacy issues without a Metropolis Hastings step. Algorithm 7 uses a CRP procedure with a very specific choice of proposal distributions for the component allocations to improve performance.

For each of the algorithms it shall be shown that the Markov Chains produced are irreducible, aperiodic and have the correct stationary distribution which is the posterior distribution of the variables in question. From these properties, the theorem 4.1 may be used to confirm that the MCMC procedure is "valid".

Algorithms 1-3 use Gibbs sampling, and therefore are guaranteed to have the correct invariant distribution.

5.1 Algorithm 1: Blackwell MacQueen Urn Scheme

Gibbs sampling for the Dirichlet Process mixture model can be performed using the conditional distributions given by the Blackwell-MacQueen urn scheme in equation (9). The state in the Markov Chain in the MCMC procedure is $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$, the collection of parameters associated with each of the n data-points $\boldsymbol{y} = (y_1, \ldots y_n)$. Algorithm 1 uses Gibbs sampling to create an ergodic chain where the limiting distribution is the distribution corresponding to density $P(\theta_{1:n}|y_{1:n})$, detailed below. At each iteration one must draw from the full conditionals, hence draw from the distribution of $\theta_i|\theta_{-i}$, $y_{1:n}$ for each $i \in \{1, \ldots, n\}$. By exchangeability, the urn scheme can be used where the i^{th} draw is viewed as the latest draw.

Algorithm 1
At each iteration:

1. For each $i \in \{1, \ldots, n\}$,draw from:

$$\theta_i|\theta_{-i},\ y_i \sim \sum_{j:j\in\{1,\ldots,n\}j\neq i} \frac{f(y_i|\theta_j)}{b}\delta_{\theta_j}(\cdot) + \frac{\alpha q_0}{b} G_{\theta|y_i}(\cdot) \tag{17}$$

where $q_0 = \int f(y_i|\theta) dG_0(\theta)$, $b = \sum_{j\neq i} f(y_i|\theta_j) + \alpha \int f(y_i|\theta) dG_0(\theta)$, and $g_{\theta|y_i}(B) = \frac{\int_B f(y_i|\theta) dG_0(\theta)}{\int f(y_i|\theta) dG_0(\theta)}$.

Invariant Distribution

The density of the invariant distribution is given by:

$$P(\theta_{1:n}|y_{1:n}) = \frac{\prod_{i=1}^{n} f(y_i|\theta_i)P(\theta_{1:n})}{\int \prod_{i=1}^{n} f(y_i|\theta_i)P(\theta_{1:n})d\theta_{1:n}}$$

Transition Kernel

Here the transition kernel density is $K(\theta_{1:n}, \theta^*_{1:n}) = \prod_{i=1}^{n} K^{(i)}(\theta_i, \theta^*_i)$, where, it is known from the stated algorithm above that:

$$K^{(i)}(\theta_i, \theta^*_i) = \frac{\sum_{j\neq i} f(y_i|\theta_j)\mathbb{I}_{\{\theta_j\}}(\theta^*_i) + \alpha q_0 g_{\theta|y_i}(\theta_{new})\mathbb{I}_{\{\theta_{new}\}}(\theta^*_i)}{b}$$

$$K^{(i)}(\theta_i, \theta^*_i) = \frac{\sum_{j\neq i} f(y_i|\theta_j)\mathbb{I}_{\{\theta_j\}}(\theta^*_i) + \alpha f(y_i|\theta_{new}) g_0(\theta_{new})\mathbb{I}_{\{\theta_{new}\}}(\theta^*_i)}{b}$$

where θ^*_i indicates the value of θ_i in the new state. Each $K^{(i)}(\theta_i, \theta^*_i)$ is dependent on the current state θ_{-i}, however, in the interest of clarity this is not displayed in $K^{(i)}$. To show that this MCMC procedure is valid, one must confirm that the kernel leads to the correct invariant distribution, that it is μ-irreducible (for some measure μ) and aperiodic.

Stationarity

In order to show that the kernel used has the desired invariant distribution, it is sufficient to show Gibbs sampling is being performed. Therefore, providing the kernel $K^{(i)}$ corresponds to the full conditional $P(\theta_i|\theta_{-i}, y_{1:n})$ then the kernel has the correct stationary distribution. In addition, by exchangeability $P(\theta_{1:n}) = P(\theta_{-i})P(\theta_i|\theta_{-i})$. Hence using the dependency strucutre given in section 3.2.:

$$\begin{aligned}
P(\theta_i|\theta_{-i}, y_{1:n}) &\propto f(y_i|\theta_i)P(\theta_{1:n}) \\
&\propto f(y_i|\theta_i)P(\theta_i|\theta_{-i}) \\
&= f(y_i|\theta_i)\left(\frac{\sum_{j:j\neq i}\mathbb{I}_{\{\theta_j\}}(\theta_i) + \alpha g_0(\theta_{new})\mathbb{I}_{\{\theta_{new}\}}(\theta_i)}{\alpha + n - 1}\right) \\
&\propto \sum_{j:j\neq i} f(y_i|\theta_i)\mathbb{I}_{\{\theta_j\}}(\theta_i) + \alpha f(y_i|\theta_{new})g_0(\theta_{new})\mathbb{I}_{\{\theta_{new}\}}(\theta_i)
\end{aligned}$$

Therefore, once the appropriate normalising constants have been re-entered it is clear that $K^{(i)}$ corresponds to the full conditional hence Gibbs sampling is being performed and the kernel leads to the correct invariant distribution.

Irreducibility and Aperiodicity

Using the same notation as in earlier sections each $\theta_i \in A, \forall i$, where A is the support of base distribution G_0. $\mu-$irreducibility must be shown. For convenience choose μ to be the base distribution G_0.
$\forall B = (B_1, \ldots, B_n)$, $\forall \theta = (\theta_1, \ldots, \theta_n)$ such that $\theta_i \in A$, $B_i \subset A$, $G_0(B_i) > 0$ $\forall i \in \{1, \ldots, n\}$:

$$\mathcal{K}(\theta, B) = \int \ldots \int_B \prod_{i=1}^{n} K^{(i)}(\theta_i, s_i)ds_1 \ldots ds_n$$

Each $K^{(i)}(\theta_i, s_i)$ is dependent on the other θ_{-i}, s_{-i}, so one cannot simply split the integral of products into products of integrals. However,

$$K^{(i)}(\theta_i, s_i) = \frac{\sum_{j\neq i} f(y_i|\theta_j)\mathbb{I}_{\{\theta_j\}}(s_i) + \alpha f(y_i|s_i)g_0(s_i)}{b} \geq \frac{\alpha f(y_i|s_i)g_0(s_i)}{b}$$

where $b = \sum_{j \neq i} f(y_i|\theta_j) + \alpha \int f(y_i|\theta) dG_0(\theta)$ and g_0 is the density of G_0.
Therefore, as each $s_i \in A = \text{support}(G_0)$, then $g(s_i) > 0$, and $\alpha f(y_i|s_i) > 0$ for a sensible choice of α and f. Hence, $\mathcal{K}(\theta, B) > 0$, and the chain is irreducible. However, notice that the sub-kernels , $K^{(i)}(\theta_i, \theta_i^*)$, clearly do not form an irreducible Markov Chain on $\theta_{1:n}$ as only parameter i is changed.

As the state in the Markov Chain, θ, may transition to any set of values in the state space A in a single transition the chain is therefore strongly irreducible and hence aperiodic. Appealing to the definition of aperiodicity given earlier, assume there exists a partition of A^n into disjoint sets $A_1 = (A_{1,1} \ldots, A_{1,n})$, $A_2 = (A_{2,1}, \ldots, A_{2,n})$, which both have positive measure under invariant distribution P, detailed above i.e. $A_{1,i} \cap A_{2,i} = \emptyset \forall i$ and $\int_{A_{j,1}} \ldots \int_{A_{j,n}} P(s_1, \ldots, s_n) ds_1 \ldots, ds_n > 0$ for $j \in \{1, 2\}$. Then, if $\theta_{1:n} \in A_1$, by the irreducibility shown above:

$$\mathcal{K}(\theta_{1:n}, A_1) > 0$$

Hence, $\mathcal{K}(\theta_{1:n}, A_2) \neq 1$ for any partition $A = A_1 \cup A_2$, so the chain is aperiodic.

Discussion

This is the simplest algorithm discussed, primarily because the state has fixed dimension, n. However, as shall be shown in section 6, this algorithm is very inefficient at sampling from the full sample space due to the state getting "stuck" in a state of high probability. This is because algorithm 1 does not facilitate a transition for the unique parameter value that is associated to multiple data-points. The discrete property of the DP means that data-points are often associated to the same parameter and are more likely to be associated to the parameter that other data-points are associated to. This is known as the "rich get richer" property. However, this may not be a parameter that has the highest likelihood for the set of data-points. Once multiple data-points are associated to this "sub-optimal" parameter, a change under algorithm 1 would require transitioning to a state of lower probability before reaching a state of overall higher probability.

5.2 Algorithm 2: Chinese Restaurant Process

As discussed in the section on Mixture Models, there is a parallel between labelling parameters for each data-point as was done in algorithm 1 with the urn representation and labelling only distinct parameter values and allocation parameters with the CRP representation. This second representation is exploited in algorithm 2 which uses the Chinese Restaurant Process procedure of first partitioning the data-points into components, i.e. sampling the Z_i, then drawing component parameters ϕ_k^* for each component $k \in \{1, \ldots, m\}$.

In this MCMC Gibbs sampling procedure, the state of the Markov Chain consists of $(Z_{1:n}, \phi^*_{1:m}, m)$. The precise values of $z_{1:n}$ does not matter providing they form some partition of the data-points into components, using integers and updating the integers based on context will be sufficient. The state-space is $C^n \times \cup_{m=1}^n (\{m\} \times A^m)$, where m is a random variable determined as the number of unique $z_{1:n}$, $m \in \mathbb{N}$, $z_i \in C = \{1, \ldots n\}, \phi_k^* \in A = \text{Support}(G_0)\ \forall i, k$. The state-space may appear quite "messy" however, it is simpler to view it as a translation of the state space from the urn representation in algorithm 1 into the CRP representaion. As there are only a finite number of data-points, only a finite number of allocations shall be used, hence it is possible to set $C = \{1, \ldots n\}$. As such, the space C can be *viewed* in terms of allocation labels associated to multiple data-points (non-singletons) and allocation labels associated to only one data-point (singleton). The label values are of no consequence. This can be exploited when considering data-point i by *viewing* as C as $\{1, \ldots, m_{-i}, m_{_{-i}+1}\}$ where m_{-i} is the number of unique z_{-i}, and $m_{-i} + 1$ denotes that z_i is a singleton. Allocation $m_{-i} + 1$ can be viewed as a new component allocation given the component allocations z_{-i}. For clarity, let $n_{-i,c}$ be the number of z_j such that $z_j = c$ and $j \neq i$, therefore if $n_{-i,z_i} = 0$ then i is a singleton.

Algorithm 2

At each iteration of the Gibbs sampler perform the following:

1. For each $i \in \{1, \ldots, n\}$
   (a) If $n_{-i,z_i} = 0$, remove $\phi^*_{z_i}$ from the list of parameters
   (b) Draw a new z_i from:

$$Z_i|\boldsymbol{\phi^*_{-i}}, \boldsymbol{z_{-i}}, \boldsymbol{y} \sim \sum_{k=1}^{m_{-i}} \frac{n_{-i,k}}{b} f(y_i|\phi_k^*)\delta_k(\cdot) + \frac{\alpha q_0}{b}\delta_{m_{-i}+1}(\cdot) \tag{18}$$

   where $q_0 = \int f(y_i|\theta) dG_0(\theta)$ and b is the appropriate scaling constant,
   (c) If z_i is not associated with any other observations, draw a new value for $\phi^*_{z_i}$ from the posterior of G_0 given observation y_i

2. Set m as the number of unique $z_{1:n}$ and re-label $z_{1:n}$ to values in the set $\{1, \ldots, m\}$ whilst preserving the partition.
   (a) For each $k \in \{1, \ldots, m\}$ draw a new value ϕ_k^* from the posterior distribution for G_0 given $\{y_i | z_i = k\ \forall i \in \{1, \ldots n\}\}$. Or some other appropriate update on $\{\phi_k^*\}_{k=1}^m$.

Invariant Distribution

In this representation, the probability density associated to each state in the stationary distribution is given by:

$$P(z_1, \ldots, z_n, \phi_1^*, \ldots, \phi_m^*|y_{1:n}) \propto \left(\prod_{i=1}^n f(y_i|\phi^*_{z_i})\right) P(z_1, \ldots, z_n, \phi_1^*, \ldots, \phi_m^*)$$

$$= \left(\prod_{i=1}^n f(y_i|\phi^*_{z_i})\right) P(\phi_1^*, \ldots, \phi_m^*) P(z_1, \ldots, z_n) = \left(\prod_{i=1}^n f(y_i|\phi^*_{z_i})\right) \left(\prod_{k=1}^m g_0(\phi_k^*)\right) \left(P(z_{-i}) \prod_{i=2}^n P(z_i|z_1, \ldots, z_{i-1})\right)$$

Transition Kernel

The transition density kernel is a cycle of kernels to update subsets of the state. Each kernel in the cycle may be generalised to a kernel on the whole state, whereby many variables transition with probability 1 to to the same values, however this complicates matters notationally. The cycle can be broken down into a Z-step corresponding to updating primarily the component allocations (though parameters may be changed if a new allocation is drawn) and a ϕ^*-step corresponding to updating the unique component parameters.

$$K\left((\phi^*_{1:m}, z_{1:n}), (\phi^{*'}_{1:m'}, z'_{1:n})\right) = K_Z\left((z_{1:n}, \phi^*_{1:m}), (z'_{1:n}, \phi^*_{1:m'})\right) K_{\phi^*}\left(\phi^*_{1:m'}, \phi^{*'}_{1:m'}\right)$$

The " ' " indicates the new value in the state. Notice how K_Z updated m to m' but not the $\{\phi_k*\}_{k=1}^m$ values.

Z-Step Kernel

K_Z is a cycle of kernels $K_Z^{(i)}$ for each i,where:
$K_Z^{(i)}\left(\left(z_i, \phi^*_{1:m^{(i-1)}}\right), \left(z'_i, \phi^*_{1:m^{(i)}}\right)\right)$

$$\propto \begin{cases} \sum_{k=1}^{m_{-i}^{(i-1)}} n_{-i,k} f(y_i|\phi_k^*)\mathbb{I}_{\{k\}}(z'_i)\left(\prod_{k=1}^{m_{-i}}\mathbb{I}_{\{\phi_k^*\}}(\phi_k^*)\right)\mathbb{I}_{\{m_{-i}^{(i-1)}\}}(m^{(i)}) & \text{non-singleton } z'_i \\ \alpha q_0 g_{\phi^*|y_i}(\phi^*_{new})\left(\prod_{k=1}^{m_{-i}^{(i-1)}}\mathbb{I}_{\{\phi_k^*\}}(\phi_k^*)\right)\mathbb{I}_{\{\phi^*_{new}\}}(\phi^*_{z_i})\mathbb{I}_{\{m_{-i}^{(i-1)}+1\}}(m^{(i)}) & \text{singleton } z'_i \end{cases} \tag{19}$$

This can be interpretted as first drawing z'_i, then given z'_i drawing $\phi^*_{1:m^{(i)}}$ as follows. Set $\phi^*_{1:m^{(i)}} = \phi^*_{1:m^{(i-1)}}$ if z'_i is a non-singleton, or introduce a new component parameter $\phi^*_{m^{(i-1)}+1}$ if a new component allocation is drawn. If $z'_i = m^{(i+1)} + 1$, a new parameter $\phi^*_{m^{(i-1)}+1} \sim G_{\phi^*|y_i}$is drawn from the distribution with density:

$$g_{\phi^*|y_i}(\phi^*_{new}) = \frac{f(y_i|(\phi^*_{new})g_0(\phi^*_{new})}{\int f(y_i|(\phi)g_0(\phi)d\phi} = \frac{f(y_i|(\phi^*_{new})g_0(\phi^*_{new})}{q_0}$$

The notation for $K_Z^{(i)}$ can be slightly misleading. $K_Z^{(i)}\left(\left(z_i, \phi^*_{1:m^{(i-1}}\right), \left(z'_i, \phi^*_{1:m^{(i)}}\right)\right)$ is not dependent on z_i, however is dependent on the other quantities of the most recent state, z_{-i}, $\phi^*_{1:m_{-i}}$. The quantity m is the number of unique components parameters, and it is important for one to remember that it is a random quantity updated indirectly through the kernel above. Notice also how the denominator in $g_{\phi^*|y_i}(\phi^*_{new})$, q_0, cancels. This is used later and motivates algorithm 8.

ϕ^*-Step Kernel

The kernel for updating $\phi^*_{1:m}$ is :

$$K_{\phi^*}(\phi^*_{1:m}, \phi^{*'}_{1:m'}) = \prod_{k=1}^{m'} K^{(k)}_{\phi^*}(\phi^*_k, \phi^{*'}_k)$$

$$K^{(k)}_{\phi^*}(\phi^*_k, \phi^{*'}_k) = g_{\phi^*|z_i, i=k, y_{1:n}}(\phi^*_k)$$

Again $K^{(k)}_{\phi^*}(\phi^*_k, \phi^{*'}_k)$ is dependent on the current state, $(z_{1:n}, \phi^*_{-k})$. To show that this MCMC procedure is valid, one must confirm that the kernel leads to the correct invariant distribution, that it is μ-irreducibility (for some measure μ) and aperiodic.

Stationarity

Unlike in algorithm 1, the dimension of the state can change. This makes it difficult notationally to establish Gibbs sampling is being used. However, it can be seen that the updates corresponding to the cycle of kernels denoted K_Z (step 1 in the algorithm) perform the exact same update as in algorithm 1 in the alternate (urn scheme) representation. Therefore, K_Z alone has the correct invariant distribution and will produce an irreducible, ergodic chain. The remaining step is to show that K_{ϕ^*} leaves the distribution invariant, then appeal to the cycle property of kernels detailed in section 4.3.

$$g_{\phi^*|z_i,y_{1:n},i=k}(\phi_k^{*'}) \propto g_0(\phi_k^{*'}) \prod_{i:z_i=k} f(y_i|\phi_k^{*'})$$

Let $T(\phi_r^*) := g_0(\phi_r^*) \prod_{i:z_i=r} f(y_i|\phi_r^*)$ therefore:

$$\begin{aligned} P(\phi_{1:m}^*, z_{1:n}|y_{1:n}) &\propto \left(\left(\prod_{i=1}^{n} f(y_i|\phi_{z_i}^*) \right) \left(\prod_{r=1}^{m} g_0(\phi_r^*) \right) (P(z_{1:n})) \right) \\ &= P(z_{1:n}) \prod_{r=1}^{m} T(\phi_r^*) \end{aligned}$$

Therefore,

$$\begin{aligned} P(\phi_{1:m}^*, z_{1:n}|y_{1:n}) K_{\phi^*}^{(k)}(\phi_k^*, \phi_k^{*'}) &\propto \left(\left(\prod_{j=1}^{n} f(y_i|\phi_{z_i}^*) \right) \left(\prod_{r=1}^{m} g_0(\phi_r^*) \right) (P(z_{1:n})) \right) \left(g_0(\phi_k^{*'}) \prod_{i:z_i=k} f(y_i|\phi_k^{*'}) \right) \\ &= P(z_{1:n}) \left(\prod_{r:r\neq k} T(\phi_r^*) \right) T(\phi_k^*) T(\phi_k^{*'}) \\ &\propto P(\phi_k^*, \phi_{-k}^*, z_{1:n}|y_{1:n}) K_{\phi^*}^{(k)}(\phi_k^{*'}, \phi_k^*) \end{aligned}$$

where the appropriate normalising constant does not depend on $\phi_k^{*'}$ or ϕ_k^*. Hence, with appropriate normalising constants, the detailed-balance equation holds:

$$P(\phi_{1:m}^*, z_{1:n}|y_{1:n}) K_{\phi^*}^{(k)}(\phi_k^*, \phi_k^{*'}) = P(\phi_k^*, \phi_{-k}^*, z_{1:n}|y_{1:n}) K_{\phi^*}^{(k)}(\phi_k^{*'}, \phi_k^*)$$

Altogether, the cycle properties of kernels means that the Markov Chain used in algorithm 2 does indeed have the correct stationary distribution.

Irreducibility and Aperiodicity

$\forall B = (B_1, \ldots, B_m)$, $\forall \phi^* = (\phi_1^*, \ldots, \phi_m^*)$ such that $\phi_k^* \in A$, $B_k \subset A$, $G_0(B_k) > 0$ $\forall k \in \{1, \ldots, m\}$:

$$\begin{aligned} \mathcal{K}_\phi(\phi^*, B) &= \int \ldots \int_B \prod_{k=1}^{m} K_{\phi^*}^{(k)}(\phi_i^*, s_i) ds_1 \ldots ds_m \\ &= \int \ldots \int_B \prod_{k=1}^{m} g_{\phi^*|z_i,i=k,y_{1:n}}(s_k) ds_1 \ldots ds_m \\ g_{\phi^*|z_i,i=k,y_{1:n}}(s_k) &= c f(y_i|s_k) g_0(s_k) > 0 \end{aligned}$$

where c is an appropriate normalising constant. $f(y_i|s_k)g_0(s_k) > 0$ as $s_k \in B_k \subset \text{Support}(G_0)$ and $f(y_i|s_k) > 0$ for sensible choice of f. Hence $\mathcal{K}(\phi^*, B) > 0$, therefore the chain $\mathcal{K}_\phi$ is strongly irreducible on the subset of the state-space that it updates, so is both irreducible and aperiodic. Hence the whole kernel, K, would be aperiodic, lead to the correct stationary distribution and irreducible on the whole state.

Discussion

It is quite difficult to articulate how this MCMC procedure works directly due to the mixtures of discrete and continuous distributions in addition to the changing dimension of the state. However, matters are simplified significantly once it is known that the updates to the state in the Markov Chain corresponding to the cycle of kernels K_Z is the same as in algorithm 1. After initialising component parameters and allocations, sampling for z_i in algorithm 2 is the same as sampling for θ_i in algorithm 1. However, the second cycle, K_{ϕ^*}, facilitates updating the parameter associated to multiple data-points which corrects for the discussed source of inefficiency in algorithm 1. Given that K_Z leads to an ergodic chain, it is not required that K_{ϕ^*} be irreducible or aperiodic, only that it has the correct invariant distribution. Therefore, many other updates can be used in its place to remedy the inefficiency of algorithm 1 or improve performance in another way.

5.3 Algorithm 3: Marginalised Chinese Restaurant Process

This algorithm marginalises out the parameters in algorithm 2, and is therefore only used when one is concerned with the partitioning of the data-points rather than the parameters of each component. This would be useful in clustering for example. This algorithm has been developed into a Hierarchical Dirichlet Process framework in section 8.

Algorithm 3

At each iteration of the Gibbs sampler perform the following:

1. For each $i \in \{1, \dots, n\}$

   (a) Draw a new z_i using the following probabilities:

$$\mathbb{P}[Z_i = z_i | z_{-i}, y_{1:n}] = \sum_{k=1}^{m_{-i}} \frac{n_{-i,k} \int f(y_i|\phi_k^*) dP_{\phi_s^*|\{y_i|z_i=k\}}(\phi_k^*)}{b} \mathbb{I}_{\{k\}}(z_i) + \frac{\alpha q_0}{b} \mathbb{I}_{\{m_{-i}+1\}}(z_i)$$

$$q_0 = \int f(y_i|\theta) dG_0(\theta)$$

Invariant Distribution

The density of the invariant distribution, up to a normalising constant c, is:

$$P(z_1, \dots, z_n | y_{1:n}) = \int \dots \int P(z_1, \dots, z_n, \phi_1^*, \dots, \phi_m^* | y_{1:n}) d\phi_1^* \dots d\phi_m^*$$

$$= c \int \dots \int \left(\prod_{j=1}^{n} f(y_j|\phi_{z_j}^*) \right) \left(\prod_{k=1}^{m} g_0(\phi_k^*) \right) (P(z_1, \dots, z_n))\, d\phi_1^* \dots d\phi_m^*$$

$$= c \int \dots \int \left(\prod_{j=1}^{n} f(y_j|\phi_{z_j}^*) \right) \left(\prod_{k=1}^{m} g_0(\phi_k^*) \right) d\phi_1^* \dots d\phi_m^* \, (P(z_1, \dots, z_n))$$

The integrands are positive, so by Fubini-Tonelli, the order of the integrals can be swapped:

$$= c \left(\prod_{k=1}^{m} \int \left(\prod_{j:z_j=k} f(y_j|\phi_k^*) \right) g_0(\phi_k^*) d\phi_k^* \right) (P(z_1, \dots, z_n))$$

where

$$c^{-1} = \int \int \left(\prod_{j=1}^{n} f(y_j|\phi_{z_j}^*) \right) \left(\prod_{k=1}^{m_{-i}} g_0(\phi_k^*) \right) (P(z_1, \dots, z_n))\, d\phi_{1:m}^* dz_{1:n}$$

Transition Kernel

The density of the transition kernel, where z_i' is the newly drawn allocation for data-point i, is:

$$K(z_{1:n}, z'_{1:n}) = \prod_{i=1}^{n} K^{(i)}(z_i, z_i')$$

$$K^{(i)}(z_i, z_i') = \sum_{k=1}^{m_{-i}} \frac{n_{-i,k} \int f(y_i|\phi_k^*) dP_{\phi_s^*|\{y_i|z_i=k\}}(\phi_k^*)}{b} \mathbb{I}_{\{k\}}(z_i') + \frac{\alpha q_0}{b} \mathbb{I}_{\{m_{-i}+1\}}(z_i')$$

where b is the appropriate normalising constant.

Stationarity

It can be seen that algorithm 3 follows algorithm 2 but integrates out the parameters $\phi^*_{1:m}$ i.e.

$$K^{(i)}(z_i, z_i') = \int K_Z^{(i)}\left(\left(z_i, \phi^*_{1:m^{(i-1)}}\right), \left(z_i', \phi^*_{1:m^{(i)}}\right)\right) d\phi^*_{1:m^{(i-1)}} d\phi^*_{1:m^{(i)}}$$

where $K_Z^{(i)}$ is given in algorithm 2. This is justified and gives the correct invariant distribution by the proof given in section 4.3 providing:

1. The invariant distribution for $Z_{1:n}$ is the marginal of that for $(Z_{1:n}, \phi^*_{1:m})$ i.e

$$P(z_1, \ldots, z_n | y_{1:n}) = \int P(z_{1:n}, \phi^*_{1:m} | y_{1:n}) d\phi^*_{1:m}$$

   which is true by choice.

2. $K_Z^{(i)}$ leads to the correct invariant distribution, $P(z_{1:n}, \phi^*_{1:m} | y_{1:n})$. This was shown to be true in algorithm 2.

Irreducibility and Aperiodicity

Similar to algorithm 2, there is a positive probability of transitioning to any state in the state space in one step, as:

$$K(z_{1:n}, z'_{1:n}) = \prod_{i=1}^{n} K^{(i)}(z_i, z_i')$$

$$K^{(i)}(z_i, z_i') > 0, \, \forall z_i, z_i' \in \{1, \ldots, m_{-i}, m_i + 1\}$$

Therefore, the chain is both irreducible and aperiodic.

5.4 Algorithm 5: Chinese Restaurant Process with Metropolis Hastings Step

When the sampling distributions $F(\cdot|\phi)$ is not conjugate to the prior G_0 it may be difficult to calculate integrals such as $\int f(y|\phi^*)dG_0(\phi^*)$ or sample directly from posterior distributions. Instead a Metropolis-Hastings step may be introduced to avoid such issues.

Algorithm 5

At each iteration of the Gibbs sampler perform the following:

1. For each $i \in \{1, \ldots, n\}$, repeat the following R times:

   (a) Draw a new Z_i from the following proposal distribution:

   $$\mathbb{P}^{'}(Z_i = c|\boldsymbol{z_{-i}}) = \begin{cases} \frac{n_{-i,c}}{n-1+\alpha} & \text{if } c \text{ is an existing component} \\ \frac{\alpha}{n-1+\alpha} & \text{if c is a new component} \end{cases} \tag{20}$$

   (b) If c is a new component, draw a value for $\phi^*_{z_i} \sim G_0$

   (c) Accept c as the new component allocation for point i with probability $\alpha(z_i, c)$, , otherwise retain the current allocation, z_i

   $$\alpha(z_i, c) = \left\{ \min 1, \frac{f(y_i|\phi^*_c)}{f(y_i|\phi^*_{z_i})} \right\}$$

2. For each $s \in \{z_1, \ldots, z_n\}$:

   (a) Draw a new value from $\phi^*_s|y_i$ such that $s = z_i$

Invariant Distribution and Transition Kernel

The invariant distribution is the same as that in algorithm 2. Similarly, the kernel can be decomposed into a Z-step and a ϕ^*-step. The ϕ^*-step is the same as algorithm 2 and the Z-step for $R = 1$ is again equivalent to algorithm 2 but with a Metropolis Hastings step within the Gibbs sampling framework to sample for each z_i. Providing the kernel corresponding to the Z-step has the correct invariant distribution, then a cycle of $R \geq 1$ such kernels will have the correct invariant distribution by the cycle property of kernels, a similar result can be shown for aperiodicity and irreducibility. Hence only the case for $R = 1$ shall be shown to have the desiered properties.

The transition density kernel for $R = 1$:

$$K\left((\phi^*_{1:m}, z_{1:n}), (\phi^{*'}_{1:m}, z'_{1:n})\right) = K_Z^{MH}\left((z_{1:n}, \phi^*_{1:m}), (z'_{1:n}, \phi^*_{1:m'})\right) K_{\phi^*}(\phi^*_{1:m'}, \phi^{*'}_{1:m'})$$

Similar to algorithm 2, K_Z^{MH} is composed of a cycle of kernels $K_Z^{MH(i)}$ for each i. $K_Z^{MH(i)}$ is a MH step targeting the $K_Z^{(i)}$given in algorithm 2 .

$$\alpha\left(\left(z_i, \phi^*_{1:m^{(i-1)}}\right), \left(z'_i, \phi^*_{1:m^{(i)}}\right)\right) = \min\left\{1, \frac{f(y_i|\phi^*_{z'_i})}{f(y_i|\phi^*_{z_i})}\right\}$$

The proposal distribution, q, is simply the prior for the allocation variables used. In the interest of readability, indicator functions are suppressed, and replaced with explanation. In addition, although $K_Z^{MH(i)}$ updates Z_i and $\phi^*_{1:m}$. The $\phi^*_{1:m}$ are only updated if a new component allocation is drawn.

Hence, in the interest of readability, reference to $\phi^*_{1:m}$ have been suppressed in the following calculations, though must still be kept in mind.
$q\left(z_i'|z_i,\right)$ is composed of drawing z_i' from:

$$\frac{\sum_{k=1}^{m_{-i}} n_{-i,k}\mathbb{I}_{\{k\}}(z_i')}{\alpha+n-1}+\frac{\alpha\mathbb{I}_{\{m_{-i}+1\}}(z_i')}{\alpha+n-1}$$

If z_i' is a new component allocation, draw $\phi^*_{z_i}\sim G_0$, otherwise $\phi^*_{z_i'}$ is the parameter of the component z_i'.
The normalising constants of the $K_Z^{(i)}$ terms cancel, similarly with the normalising terms for the proposal q. In addition the α or $n_{-i,k}$ terms will also cancel between $K_Z^{(i)}(z_i,z_i')$ and $q(z_i'|z_i)$ etc. For example, consider the case where z_i' is a singleton, z_i non-singleton

$$\begin{aligned}\alpha\left(z_i,z_i'\right) &= \min\left\{1,\frac{K_Z^{(i)}(z_i,z_i')q(z_i|z_i')}{K_Z^{(i)}(z_i',z_i)q(z_i'|z_i)}\right\}\\ &= \min\left\{1,\frac{\alpha g(\phi^*_{z_i'})f(y_i|\phi^*_{z_i'})}{n_{-i,z_i}f(y_i|\phi^*_{z_i})}\frac{n_{-i,z_i}}{\alpha g(\phi^*_{z_i'})}\right\}\\ &= \min\left\{1,\frac{f(y_i|\phi^*_{z_i'})}{f(y_i|\phi^*_{z_i})}\right\}\end{aligned}$$

The other cases follow similarly.

Stationarity, Irreducibility and Aperiodicity of the Z-step

The kernel performs Metropolis Hastings updates within a Gibbs sampling framework, hence leads to the correct invariant distribution by design. The Gibbs sampling framework is the same as in algorithm 2 and the support of the proposal distribution is the same as the support of the target in the MH steps as the proposal is the prior distribution of the target. Therefore, the kernel for the Z-step is irreducible and aperiodic for the component allocation variables. As shown in algorithm 2, the Z-step is also irreducible and aperiodic for the component parameters given it mimics algorithm 1 in the CRP representation.

Discussion

This algorithm is essentially the same as algorithm 2 with a Metropolis Hastings step to update component allocations using the proposal distribution as the conditional prior of component allocations (i.e the distribution of component allocations not given the data). Given that the kernel for updating the component allocations has the correct invariant distribution, it can be repeated R times and this cycle will have the correct invariant distribution. This Metropolis Hastings step is used if $\int f(y_i|\phi)dG_0(\phi)$ cannot be computed, however a very similar requirement is needed to sample directly from the posterior using kernel K_{ϕ^*}. Therefore, in practice this last step may also require a Metropolis Hastings step, or something similar, as shall be done in the application to classification in section 7.

5.5 Algorithm 6: Urn Scheme with Metropolis Hastings Step

Algorithm 6

At each iteration of the Gibbs sampler perform the following:

1. For each $i \in \{1, \ldots, n\}$, repeat the following R times:

   (a) Draw a new θ_i' from the following proposal distribution:

   $$\frac{1}{n-1+\alpha}\sum_{i=1}^{n}\delta_{\theta_i}(\cdot) + \frac{\alpha}{n-1+\alpha}G_0(\cdot)$$

   (b) Accept θ_i' as the new component parameter for point i with probability $\alpha(\theta_i, \theta_i')$, otherwise retain the previous value θ_i

   $$\alpha(\theta_i, \theta_i') = \min\left\{1, \frac{f(y_i|\theta_i')}{f(y_i|\theta_i)}\right\}$$

Invariant Distribution and Transition Kernel

Algorithm 6 uses the urn representation, and hence has the same invariant distribution as in algorithm 1. It is clear that algorithm 6 is uses the same Gibbs sampling procedure as algorithm 1 with a Metropolis Hastings step to sample from the full conditionals for each i. The MH step is repeated R times for each i, however this does not pose an issue due to the cycle property of kernels. The proposal distribution for the element-wise MH step is the conditional prior for the parameters θ_i given below as:

$$q(\theta_i'|\theta_i) = \frac{1}{n-1+\alpha}\sum_{j\neq i}\mathbb{I}_{\{\theta_i\}}(\theta_i') + \frac{\alpha}{n-1+\alpha}g_0(\theta_i')$$

This is essentially the same proposal in the Z-step of algorithm 5 translated into the urn representation. Also, given it is the prior, it is clear to see without further justification that the acceptance probability is as before:

$$\alpha(\theta_i, \theta_i') = \min\left\{1, \frac{f(y_i|\theta_i')}{f(y_i|\theta_i)}\right\}$$

Stationarity, Irreducibility and Aperiodicity

Stationarity, irreducibility and aperiodicity are guaranteed by the same justification in algorithm 1, the extra complexity from the MH step will not pose a problem by the same argument as in algorithm 5 i.e. using the prior as a proposal distribution to sample from the posterior in the MH step will result in a strongly irreducible chain.

Discussion

This algorithm is the same as algorithm 1 but with a Metropolis Hastings step using the prior as the proposal distribution. It suffers from the same source of inefficiency as algorithm 1 in that it does not facilitate a block update of $\{\theta_i|\theta_i = \phi_k^*\}$ for each component k.

5.6 Algorithm 7: Metropolis Hastings with Specific Proposal

Algorithm 7 follows a similar framework to algorithm 2 and 5 in that it uses the CRP representation split into a $Z-$step and a ϕ^*-step. The ϕ^*-step (Step 3) is the same as in previous algorithms. The added complexity is in the Z-step consisting of a cycle of two kernels. The first corresponds to Step 1 below, and updates allocation z_i following a Metropolis Hastings step within the Gibbs sampling framework of algorithm 2, whereby the the proposal distribution depends on whether z_i is a singleton ($n_{-i,z_i} = 0$) or not ($n_{-i,z_i} > 0$). This is very similar to algorithm 5 but with quite a specific proposal. The second kernel (Step 2) is only a partial update for the non-singleton allocations to other non-singleton allocations element wise.

Algorithm 7

At each iteration:

1. For $i = 1, \ldots, n$:
   (a) If $n_{-i,z_i} > 0$:
      i. Let c be a new component i.e. $\max_i(z_i) + 1$
      ii. Draw ϕ_c^* from:
      $$\phi_c^* \sim G_0$$
      iii. Set $z_i = c$ with probability $\alpha(z_i, c)$, otherwise z_i does not change
      $$\alpha_1(z_i, c) = \min\left\{1, \frac{\alpha}{n-1}\frac{f(y_i|\phi_c^*)}{f(y_i|\phi_{z_i}^*)}\right\}$$
      iv. If a new component allocation is selected:
         A. Add ϕ_c^* to the state
         B. Set m as the new number of of unique component allocations
   (b) If $n_{-i,z_i} = 0$:
      i. Draw c from a list of the existing component allocations $\{1, \ldots, m_{-i}\}$ with probabilities:
      $$\mathbb{P}[c = k] = \frac{n_{-i,k}}{n-1} \; k \in \{1, \ldots m\}$$
      ii. Set $Z_i = c$ with probability $\alpha(z_i, c)$, otherwise z_i does not change
      iii. If a new component allocation is selected m as the new number of of unique component allocations
2. For $i \in \{1, \ldots, n\}$:
   (a) If $n_{-i,z_i} > 0$:
      i. Set $Z_i$to k from a list of the existing component allocations with probabilities:
      $$\mathbb{P}[Z_i = k] \propto \frac{n_{-i,k}}{n-1} f(y_i|\phi_k^*) \; k \in \{1, \ldots m\}$$
3. For each $k \in \{z_1, \ldots, z_n\}$:
   (a) Draw a new value from $\phi_s^*|\{y_i \text{ such that } z_i = k\}$ or any other update that leaves the distribution invariant

Invariant Distribution and Transition Kernel

The target distribution is the same as given in algorithm 2 as the CRP representation is used. The transition kernel can be decomposed into the Z-step and the ϕ^*-step, the latter is the same as in algorithm 2. The Z-step kernel, K_Z^7, is a cycle of two kernels K_1 and K_2, which are both in turn cycles of element-wise kernels denoted $K_1^{(i)}$,$K_2^{(i)}$ for each $i \in \{1, \dots n\}$

$$K\left(\left(z_{1:n}, \phi^*_{1:n}\right), \left(z'_{1:n}, \phi^{*'}_{1:n}\right)\right) = K_Z^7\left(\left(z_i, \phi^*_{1:m}\right), \left(z'_i, \phi^*_{1:m'}\right)\right) K_{\phi^*}(\phi^*_{1:m}, \phi^{*'}_{1:m})$$

Kernel-$K_1^{(i)}$

This is a MH step as detailed above with acceptance probability $\alpha_1(\cdot,\cdot)$, and proposal distribution q_1 detailed below. Similar to algorithm 5, the target distribution is the full conditional for the component allocation variable, denoted $K_Z^{(i)}$ in algorithm 2.

- The proposal distribution is:
  $q_1\left(\left(z'_i, \phi^*_{1:m'}\right) \middle| \left(z_i, \phi^*_{1:m}\right)\right) =$

$$\begin{cases} \mathbb{I}_{\{k_{new}\}}(z'_i) g_0(\phi^*_{new}) \left(\prod_{k=1}^{m_{-i}} \mathbb{I}_{\{\phi^*_k\}}(\phi^*_k)\right) \mathbb{I}_{\{\phi^*_{new}\}}(\phi^*_{z_i}) \mathbb{I}_{\{m_{-i}+1\}}(m') & \text{if } z_i\text{not a singleton i.e } n_{-i,z_i} > 0 \\ \frac{\sum_{k=1}^{m_{-i}} n_{-i,k} \mathbb{I}_{\{k\}}(z'_i)}{n-1} \left(\prod_{k=1}^{m_{-i}} \mathbb{I}_{\{\phi^*_k\}}(\phi^*_k)\right) \mathbb{I}_{\{m_{-i}\}}(m') & \text{if } z_i\text{is a singleton i.e } n_{-i,z_i} = 0 \end{cases}$$

- The acceptance probability is:

$$\alpha_1\left(\left(z_i, \phi^*_{1:m}\right), \left(z'_i, \phi^*_{1:m'}\right)\right) = \begin{cases} \min\left\{1, \frac{\alpha}{n-1} \frac{f(y_i|\phi^*_{z'_i})}{f(y_i|\phi^*_{z_i})}\right\} & \text{if } z_i\text{not a singleton i.e } n_{-i,z_i} > 0 \\ \min\left\{1, \frac{n-1}{\alpha} \frac{f(y_i|\phi^*_{z'_i})}{f(y_i|\phi^*_{z_i})}\right\} & \text{if } z_i\text{is a singleton i.e } n_{-i,z_i} = 0 \end{cases}$$

  The acceptance probabilities can be shown to be correct in two cases. Again I suppress reference to $\phi^*_{1:m}$ for clarity

1. z_i not a singleton, z'_i singleton

$$\begin{aligned} \alpha_1\left(\left(z_i, \phi^*_{1:m}\right), \left(z'_i, \phi^*_{1:m'}\right)\right) &= \min\left\{1, \frac{K_Z^{(i)}(z_i, z'_i) q_1\left(z_i | z'_i\right)}{K_Z^{(i)}(z'_i, z_i) q_1\left(z'_i | z_i\right)}\right\} \\ &= \min\left\{1, \frac{\alpha g(\phi^*_{z'_i}) f(y_i|\phi^*_{z'_i}) \left(\frac{n_{-i,z_i}}{n-1}\right)}{n_{-i,z_i} f(y_i|\phi^*_{z_i}) g_0(\phi^*_{z'_i})}\right\} \\ &= \min\left\{1, \frac{\alpha}{n-1} \frac{f(y_i|\phi^*_{z'_i})}{f(y_i|\phi^*_{z_i})}\right\} \end{aligned}$$

2. z_i is a singleton, z'_i not a singleton

$$\begin{aligned} \alpha_1\left(\left(z_i, \phi^*_{1:m}\right), \left(z'_i, \phi^*_{1:m'}\right)\right) &= \min\left\{1, \frac{K_Z^{(i)}(z_i, z'_i) q_1\left(z_i | z'_i\right)}{K_Z^{(i)}(z'_i, z_i) q_1\left(z'_i | z_i\right)}\right\} \\ &= \min\left\{1, \frac{n_{-i,z'_i} f(y_i|\phi^*_{z'_i}) g_0(\phi^*_{z_i})}{\alpha g(\phi^*_{z_i}) f(y_i|\phi^*_{z_i}) \left(\frac{n_{-i,z'_i}}{n-1}\right)}\right\} \\ &= \min\left\{1, \frac{n-1}{\alpha} \frac{f(y_i|\phi^*_{z'_i})}{f(y_i|\phi^*_{z_i})}\right\} \end{aligned}$$

Kernel-$K_2^{(i)}$

This kernel facilitates changing component allocations for only the non-singletons to other non-singleton allocations. The singleton allocations are not changed.

$$K_2^{(i)}(z_i, z_i') = \begin{cases} \frac{\sum_{k=1}^{m} n_{-i,k} f(y_i|\phi_k^*)\mathbb{I}_k(z_i')}{n-1} & \text{if } z_i \text{ not a singleton i.e } n_{-i,z_i} > 0 \\ \mathbb{I}_{\{z_i\}}(z_i') & \text{if } z_i \text{ is a singleton i.e } n_{-i,z_i} = 0 \end{cases}$$

Stationarity: Kernel

Each kernel $K_1^{(i)}$ uses MH-steps and therefore has the correct invariant distribution. Kernel, $K_2^{(i)}$, can be understood as a partial Gibbs sampling step, as Gibbs sampling is performed on a restriction of the state including just the allocation variables which share a component allocation with another data-point. Detailed balance will be demonstrated to show the desired invariance.

$$P(z_1, \ldots, z_n, \phi_1^*, \ldots, \phi_m^*|y_{1:n})K_2^{(i)}(z_i, z_i') = K_2^{(i)}(z_i', z_i)P(z_i', z_{-i}, \phi_{1:m}^*|y_{1:n})$$

- Case 1: z_i is a singleton

$$\begin{aligned} P(z_1, \ldots, z_n, \phi_1^*, \ldots, \phi_m^*|y_{1:n})K_2^{(i)}(z_i, z_i') &= P(z_1, \ldots, z_n, \phi_1^*, \ldots, \phi_m^*|y_{1:n})\mathbb{I}_{\{z_i\}}(z_i') \\ &= \begin{cases} 0 & \text{if } z' \neq z_i \\ P(z_1', \ldots, z_n, \phi_1^*, \ldots, \phi_m^*|y_{1:n})\mathbb{I}_{\{z_i'\}}(z_i) & \text{if } z' = z_i \end{cases} \end{aligned}$$

  Hence,

$$P(z_1, \ldots, z_n, \phi_1^*, \ldots, \phi_m^*|y_{1:n})\mathbb{I}_{\{z_i\}}(z_i') = P(z_1', \ldots, z_n, \phi_1^*, \ldots, \phi_m^*|y_{1:n})\mathbb{I}_{\{z_i'\}}(z_i)$$

- Case 2: z_i not a singleton
  Let

$$c = \frac{\left(\prod_{j:j\neq i} f(y_j|\phi_{z_j}^*)\right)\left(\prod_{k=1}^{m} g_0(\phi_k^*)\right)P(z_{-i})}{\int\ldots\int\left(\prod_{j=1}^{n} f(y_j|\phi_{z_j}^*)\right)\left(\prod_{k=1}^{m^{(i-1)}} g_0(\phi_k^*)\right)(P(z_1, \ldots, z_n))\, d\phi_{1:m}^* dz_{1:m}}$$

  Then:

$$\begin{aligned} P(z_{1:n}, \phi_{1:m}^*|y_{1:n})K_2^{(i)}(z_i, z_i') &= cf(y_i|\phi_{z_i}^*)\left(P(z_i|z_{-i})\right)\left(\frac{\sum_{k=1}^{m_{-i}} n_{-i,k} f(y_i|\phi_k^*)\mathbb{I}_{\{k\}}(z_i')}{n-1}\right) \\ &= cf(y_i|\phi_{z_i}^*)\left(\frac{\sum_{k=1}^{m_{-i}} n_{-i,k}\mathbb{I}_{\{k\}}(z_i)}{\alpha+n-1}\right)\left(\frac{\sum_{k=1}^{m_{-i}} n_{-i,k} f(y_i|\phi_{z_i'}^*)\mathbb{I}_{\{k\}}(z_i')}{n-1}\right) \\ &= c\left(\frac{\sum_{k=1}^{m_{-i}} f(y_i|\phi_{z_i}^*)n_{-i,k}\mathbb{I}_{\{k\}}(z_i)}{n-1}\right)\left(\frac{\sum_{k=1}^{m_{-i}} n_{-i,k}\mathbb{I}_{\{k\}}(z_i')}{\alpha+n-1}\right) f(y_i|\phi_{z_i'}^*) \\ &= cK_2^{(i)}(z_i', z_i)P(z_i'|z_{-i})f(y_i|\phi_{z_i'}^*) \\ &= K_2^{(i)}(z_i', z_i)P(z_i', z_{-i}, \phi_{1:m}^*|y_{1:n}) \end{aligned}$$

  Hence:

$$P(z_1, \ldots, z_n, \phi_1^*, \ldots, \phi_m^*|y_{1:n})K_2^{(i)}(z_i, z_i') = K_2^{(i)}(z_i', z_i)P(z_i', z_{-i}, \phi_{1:m}^*|y_{1:n})$$

  Therefore, detailed balance holds, $K_2^{(i)}$ is reversible, so the chain with kernels $\{K_2^{(i)}\}_{i=1}^n$ has the correct invariant distribution by the cycle property.

Irreducibility and Aperiodicity

Irreducibility and aperiodicity was justified by [3] by noticing "there is a nonzero probability that a single scan of the data items will result in a state where every data item is associated with a different component". It is difficult to show for the general case that the chain generated is both irreducible and aperiodic as the support of proposal distribution for kernels $\{K_1^{(i)}\}_i$ is a strict subset of the support of the target distribution. The proposal distribution gives zero probability of a singleton transitioning to another singleton, and zero probability of a non-singleton transitioning to a non-singleton, however the non-singleton to non-singleton case is catered for in $K_2^{(i)}$. The restricted support of the proposal would not be an issue in the MH step if the acceptance probability is less than 1 for the "singleton to non-singleton" transition as the only transition not catered for is "singleton to singleton" which is equivalent to no transition. The acceptance probability for "singleton to non-singleton" is $\min\left\{1, \frac{n-1}{\alpha}\frac{f(y_i|\phi^*_{z_i'})}{f(y_i|\phi^*_{z_i})}\right\}$, which is dependent on the likelihood function $f(\cdot|\cdot)$. It is clear to see that it possible for a rejected proposal using the normal distribution, however, it is not immediately obvious that this would be the case for any likelihood function, especially as one would expect the number of data-points, n, to be significantly larger than α. If the likelihood function was uniform i.e. constant and $n-1 > \alpha$ then the acceptance probability would always be 1.

5.7 Algorithm 8: Auxiliary Variable Method

Algorithm 8 uses auxiliary variables to circumnavigate difficult integrals. Instead of drawing a new component allocation then drawing a parameter for this new allocation, a new component parameter is drawn (actually M parameters are independently drawn) from the base distribution, then the component allocation is drawn given these parameters. It is shown below that such an algorithm is indeed valid. The new M new parameters are treated as auxiliary variables that only exist temporarily. The permanent state in the Markov Chain consists of $(Z_{1:n}, \phi^*_{1:m})$ where m is set to the number of unique component allocations in $Z_{1:n}$.

Algorithm 8

At each iteration of the Gibbs sampler perform the following:

1. For each $i \in \{1, \ldots, n\}$: Let m_{-i} be the number of unique component allocations excluding z_i, and allocate each component parameter an allocation in $\{1, \ldots m_{-i}\}$.

   (a) If z_i is a singleton, set $\varphi_1^{(i)} = \phi^*_{z_i}$ else draw $\varphi_1^{(i)} \sim G_0$

   (b) Draw $(\varphi_2^{(i)}, \ldots, \varphi_M^{(i)})$ from
   $$\varphi_j^{(i)} \sim G_0$$
   $\{\varphi_j^{(i)}\}_{j=1}^M$ are drawn independently

   (c) Draw a new z_i from the following proposal distribution:
   $$\mathbb{P}(Z_i = z_i' | z_{-i}, \phi^*_{1:m}, \varphi^{(i)}_{1:M}) \propto \sum_{k=1}^{m_{-i}} n_{-i,k} f(y_i|\phi^*_k)\mathbb{I}_{\{k\}}(z_i') + \frac{\alpha}{M} \sum_{j=m_{-i}+1}^{m_{-i}+M} f(y_i|\varphi^{(i)}_{j-m_{-i}})\mathbb{I}_{\{j\}}(z_i') \tag{21}$$

   (d) If the newly drawn $z_i \in \{m_{-i}+1, \ldots, m_{-i}+M\}$ then:

      i. Set m as $m_{-i}+1$

      ii. Set z_i as $m_{-1}+1$

      iii. Set $\phi^*_{z_1}$ as the parameter $\varphi_j^{(i)}$ corresponding to the draw in (21)

2. For each $k \in \{z_1, \ldots, z_n\}$:

   (a) Draw a new value from $\phi^*_k | \{y_i \text{ such that } z_i = k\}$, or some other appropriate update on $\{\phi^*_k\}_{k=1}^m$.

In step 1, either $M-1$ or M auxiliary variables are drawn depending on whether z_i is a singleton or not.

Invariant Distribution

The permanent state in the Markov Chain is $z_1, \ldots, z_n, \phi^*_1, \ldots, \phi^*_m$ with target distribution the same as given in algorithm 2:

$$\begin{aligned} P(z_{1:n}, \phi^*_{1:m}|y_{1:n}) &\propto \left(\prod_{i=1}^n f(y_i|\phi^*_{z_i})\right) P(z_1, \ldots, z_n, \phi^*_1, \ldots, \phi^*_m) \\ &= \left(\prod_{i=1}^n f(y_i|\phi^*_{z_i})\right) P(\phi^*_1, \ldots, \phi^*_m) P(z_1, \ldots, z_n) \\ &= \left(\prod_{i=1}^n f(y_i|\phi^*_{z_i})\right) \left(\prod_{k=1}^m g_0(\phi^*_k)\right) (P(z_{-i})P(z_i|z_{-i})) \end{aligned}$$

When considering allocation i, the extended state in the Markov Chain, with temporary variables $\varphi_1^{(i)}, \ldots, \varphi_M^{(i)}$, is:

$$z_1, \ldots, z_n, \phi_1^*, \ldots, \phi_{m_{-i}}^*, \varphi_1^{(i)}, \ldots, \varphi_M^{(i)}$$

where $\varphi_1^{(i)} = \phi_{z_i}^*$ if z_i is a singleton.
Let:

$$M(z_i) = \begin{cases} \{1, \ldots, M\} & \text{if } z_i\text{is not a singleton} \\ \{2, \ldots, M\} & \text{if } z_i\text{is a singleton} \end{cases}$$

The target distribution of the augmented state is as follows, where c_1 is the appropriate normalising constant. :

$$\begin{aligned} P(z_{1:n}, \phi_{1:m}^*, \varphi_{1:M}^{(i)} | y_{1:n}) &= c_1 \left(\prod_{j=1}^{n} f(y_j | \phi_{z_j}^*) \right) \left(\prod_{k=1}^{m_{-i}} g_0(\phi_k^*) \right) \left(\prod_{r=1}^{M} g_0(\varphi_r^{(i)}) \right) (P(z_{-i}) P(z_i | z_{-i})) \\ &= c_1 \left(\prod_{j=1}^{n} f(y_j | \phi_{z_j}^*) \right) \left(\prod_{k=1}^{m} g_0(\phi_k^*) \right) \left(\prod_{r \in M(z_i)} g_0(\varphi_r^{(i)}) \right) (P(z_{-i}) P(z_i | z_{-i})) \end{aligned}$$

Let $c_2 := \left(c_1 \prod_{j:j \neq i} f(y_j | \phi_{z_j}^*) \prod_{k=1}^{m_{-i}} g_0(\phi_k^*) P(z_{-i}) \right)$, therefore:

$$P(z_{1:n}, \phi_{1:m}^*, \varphi_{1:M}^{(i)} | y_{1:n}) = c_2 f(y_i | \phi_{z_i}^*) P(z_i | z_{-i}) \prod_{r=1}^{M} g_0(\varphi_r^{(i)})$$

As before, the conditional prior on component allocations is:

$$P(z_i | z_{-i}) \propto \begin{cases} \alpha & \text{if } z_i\text{is a new component allocation} \\ n_{-i,k} & \text{if } z_i = k\text{, an existing component allocation} \end{cases}$$

If a new component allocation is drawn, then a a new component parameter is drawn. However, in the extended state view this can be viewed as drawing M new component parameters. When a new component allocation is drawn, the new associated component parameter can be chosen uniformly from the M drawn component parameters. This gives rise to the same prior on $z_i | z_{-i}$, however, the drawn parameter was chosen with probability $\frac{1}{M}$ from the M drawn. This is a very subtle difference to the other algorithms that does not pose an issue when running the algorithm as the $\frac{1}{M}$ cancels in the normalising constant. However, it is important to acknowledge this term to establish detailed balance.

Transition Kernel

Step 1 performs the Z-step update by performing an update on the extended state then recording only the values for the variables of interest, $z_{1:n}, \phi_{1:m}^*$. Step 2 in the algorithm is the same as in algorithm 2, corresponding to kernel K_{ϕ^*}. Let $\varphi = \{\varphi_{M(z_1)}^{(1)}, \ldots, \varphi_{M(z_n)}^{(n)}\}$, then the kernel for the extended state is:

$$K\left((z_{1:n}, \phi_{1:m}^*, \varphi), \left(z'_{1:n}, \phi_{1:m'}^{*'}, \varphi \right) \right) = K_{Aug}\left((z_{1:n}, \phi_{1:m}^*, \varphi), (z'_{1:n}, \phi_{1:m'}^{*'}, \varphi) \right) K_{\phi^*}(\phi_{1:m'}^*, \phi_{1:m'}^{*'})$$

Reference to φ has been suppressed for clarity.
K_{Aug} is a cycle of n kernels denoted $K_{Aug}^{(i)}$ for each $i \in \{1, \ldots, n\}$. The auxiliary variables, $\varphi_{1:M(z_i)}^{(i)}$, are used but not updated.

$$\begin{aligned} &K_{Aug}^{(i)}\left(\left(z_i, \phi_{1:m}^*, \varphi_{1:M(z_i)}^{(i)} \right), \left(z'_i, \phi_{1:m^{(i)}}^*, \varphi_{1:M(z_i)}^{(i)} \right) \right) \\ &\propto \begin{cases} \sum_{k=1}^{m_{-i}} n_{-i,k} f(y_i | \phi_k^*) \mathbb{I}_{\{k\}}(z'_i) \mathbb{I}_{\{m_{-i}\}}(m^{(i)}) & \text{if } z'_i \text{ non-singleton} \\ \frac{\alpha}{M} \sum_{j=1}^{M} f(y_i | \varphi_j^{(i)}) \mathbb{I}_{\{m_{-i}+1\}}(z'_i) \mathbb{I}_{\{m_{-i}+1\}}(m^{(i)}) \mathbb{I}_{\{\varphi_j^{(i)}\}}(\phi_{m'}^*) & \text{if } z'_i \text{ a new component allocation} \end{cases} \end{aligned} \tag{22}$$

Stationarity

To establish stationarity, it is sufficient to establish detailed balance:

$$P(z_i, z_{-i}, \phi^*_{1:m}, \varphi^{(i)}_{M(z_i)}|y_{1:n}) K^{(i)}_{Aug}\left(\left(z_i, \phi^*_{1:m}, \varphi^{(i)}_{1:M(z_i)}\right), \left(z_i', \phi^*_{1:m'}, \varphi^{(i)}_{1:M(z_i)}\right)\right)$$

$$= P(z_i', z_{-i}, \phi^*_{1:m'}, \varphi^{(i)}_{M(z_i)}|y_{1:n}) K^{(i)}_{Aug}\left(\left(z_i', \phi^*_{1:m'}, \varphi^{(i)}_{1:M(z_i)}\right), \left(z_i, \phi^*_{1:m}, \varphi^{(i)}_{1:M(z_i)}\right)\right)$$

Although slightly tedious, this can be established in 4 cases depending on whether z_i and z_i' are singletons or not (the detailed algebra is given in the appendix). Therefore, given kernels $\{K^{(i)}_{Aug}\}_i$ each have the correct invariant distribution, by the properties of cycles, K_{Aug} has the correct invariant distribution. K_{ϕ^*} is the same as in algorithm 2 so has the correct invariant distribution. The next step in verifying this procedure is valid is to show that the marginal distribution of $P(z_i, z_{-i}, \phi^*_{1:m}\varphi^{(i)}_{M(i)}|y_{1:n})$ is $P(z_{1:n}, \phi^*_{1:m}|y_{1:n})$.

$$\begin{aligned}
\int P(z_i, z_{-i}, \phi^*_{1:m}\varphi^{(i)}_{M(z_i)}|y_{1:n}) d\varphi^{(i)}_{M(z_i)} &= \int P(\varphi^{(i)}_{M(i)}|y_{1:n}, z_i, z_{-i}, \phi^*_{1:m}) P(z_{1:n}, \phi^*_{1:m}|y_{1:n}) d\varphi^{(i)}_{M(z_i)} \\
&= \int \left(\prod_{r\in M(z_i)} g_0(\varphi^{(i)}_r)\right) P(z_{1:n}, \phi^*_{1:m}|y_{1:n}) d\varphi^{(i)}_{M(z_i)} \\
&= P(z_{1:n}, \phi^*_{1:m}|y_{1:n}) \prod_{r\in M(z_i)} \left(\int g_0(\varphi^{(i)}_r) d\varphi^{(i)}_r\right) \\
&= P(z_{1:n}, \phi^*_{1:m}|y_{1:n})
\end{aligned}$$

as $\int \left(\prod_{r\in M(i)} g_0(\varphi^{(i)}_r)\right) d\varphi^{(i)}_{M(i)} = 1$ and where c_1 is the appropriate normalising constant.

$$c_1^{-1} = \int \left(\prod_{j=1}^{n} f(y_j|\phi^*_{z_j})\right) \left(\prod_{k=1}^{m} g_0(\phi^*_k)\right) \left(P(z_{-i})P(z_i|z_{-i})\right) d\phi^*_{1:m}$$

Irreducibility and Aperiodicity

It can be seen that , for each state z_i $\forall i \in \{1, \ldots, n\}$, there is a positive probability of transitioning to any other state in $\{1, \ldots, m_{-i}, m_{-1}+1\}$. In the extended state, the transition kernel corresponding to updating z_i is: $K^{(i)}_{Aug}\left(\left(z_i, \phi^*_{1:m}, \varphi^{(i)}_{1:M(z_i)}\right), \left(z_i', \phi^*_{1:m^{(i)}}, \varphi^{(i)}_{1:M(z_i)}\right)\right)$ detailed in line 22. Each parameter $\varphi^{(i)}_r \in \text{support}(G_0)$, and assume $f(y_i|\phi^*_k) > 0 \forall k \in \{1, \ldots, m_{-i}, m_{-1}+1\}$. Then for any drawn $z_i' \in \{1, \ldots, m_{-i}, m_{-i}+1\}$:

$$K^{(i)}_Z\left(\left(z_i, \phi^*_{1:m}, \varphi^{(i)}_{1:M(z_i)}\right), \left(z_i', \phi^*_{1:m^{(i)}}, \varphi^{(i)}_{1:M(z_i)}\right)\right) > 0$$

Hence there is positive probability for each z_i transitioning to any other allocation label in a single transition. Therefore the cycle of kernels corresponding to updating $Z_{1:n}$ is irreducible and aperiodic. $K_{\phi^*}(\phi^*_{1:m}, \phi^{*'}_{1:m})$ is the same as in algorithm 2, and has been shown to be aperiodic and irreducible on the subset of the state space corresponding to parameters $\phi^*_{1:m}$. The cycle of chains K_{Aug} and K_{ϕ^*} is aperiodic and irreducible, hence algorithm 8 is valid for taking approximate samples from the posterior of the parameters.

Discussion

The performance of algorithm 8 depends heavily on the choice of M. A large M would approximate algorithm 2, in the sense that $\frac{\alpha}{M}\sum_{j=1}^{M} f(y_i|\varphi^{(i)}_j) g(\varphi^{(i)}_j)$ is a Monte Carlo approximation of $\int f(y_i|\phi) d(\phi)$

as the $\varphi_j^{(i)}$ are independent and identically distributed. The cost of this is the extra computational time. The benefit of algorithm 8 over the others is that there is no Metropolis Hastings step, and therefore computational effort is not wasted rejecting proposals or repeating cycles of kernels to avoid high autocorrelations (to be explained later).

6 Algorithm Evaluation

6.1 Neal's 9 Data Points

In the [3] paper, the non-conjugate algorithms were investigated, in particular, time per iteration and autocorrelation time[27, 3] were used to judge the performance of each algorithm. Although efficiency and speed are very important factors in assessing the performance of each algorithm, time per iteration was not considered here. Even after controlling for computer processing power, time per iteration is largely dependent on the efficiency of the implementation of each algorithm which is difficult to control for. Autocorrelation time can be thought of as the factor in which the sample size has been reduced by taking correlated samples from the MCMC procedure as opposed to taking independent draws from the target distribution. This was estimated using [27] in the results below. More details are given in [37].

I repeated Neal's experiment in order to check my implementations are working as expected. The results were very similar to those of Neal [3], which is indicative that the implementations do not have major errors. However, in addition to investigating the non-conjugate algorithms (5, 6 ,7 , 8) carried out in [3], I also looked into the conjugate algorithms (1 and 2) to compare performance.

Neal's 9 data-points are:

$$\{x_i\}_{i=1}^{9} = (-1.48,\ -1.40,\ -1.16,\ -1.08,\ -1.02,\ +0.14,\ +0.51,\ +0.53,\ +0.78)$$

The model used is:

$$\begin{aligned} G &\sim DP(\alpha, G_0) & \alpha &= 1 \\ & & G_0 &= N(0,1) \\ \theta_i|G &\sim G & F(\cdot,\theta) &= N(\theta, 0.1^2) \\ x_i &\sim F(\cdot, \theta_i) \end{aligned}$$

The Gibbs sampling step corresponding to updating the component parameters given the component allocations and data, i.e. the step corresponding to kernel $K_{\phi^*}(\phi^*_{1:m}, \phi^{*'}_{1:m'})$ in algorithms 2,5,7,8, was carried out using the posterior distribution by taking advantage of the conjugacy between F and G_0. This could have been done through a Metropolis Hastings step however, given this update is not the main focus of the report the posterior was used for convenience. Note that even though F and G_0 are conjugate, they are treated as non-conjugate for the non-conjugate algorithms other than in the aforementioned step of updating the parameters.

The initialisation consisted of a uniformly random clustering and a burn-in period of 100 iterations. The values of θ_1 and the number of clusters, m, were recorded over 20,000 iterations.

Algorithm	Autocorrelation Time for m	Autocorrelation Time for θ_1
1	∞(constant m)	27.249
2	1.7160	1.9882
5 ($R = 4$)	8.2640	9.1672
6 ($R = 4$)	17.8004	71.7818
7	6.9357	5.1990
8 ($M = 1$)	4.1287	6.9716
8 ($M = 2$)	3.1003	5.0778
8 ($M = 30$)	1.9481	2.6025

Table 3: Neal's 9 Data-points

There are two main sources of correlation in the MCMC procedures. Firstly when accepted proposals are "close" to the existing state there will be correlation, and secondly in the case of Metropolis Hastings if the proposals are rejected then there will be autocorrelation as the recorded state from one iteration to the next will be the same. Therefore, as algorithms 1 and 2 do not use MH steps, it is as expected that these algorithms exhibit lower autocorrelation times than the MH versions (algorithms 5,6). Though the autocorrelation times may be reduced for algorithms 5 an 6 by increasing the number of cycles of updated per recorded iteration by increasing R. Algorithm 8 does not use MH steps, however algorithm 7 does. Therefore it is noteworthy that algorithm 7 has a lower autocorrelation time than for algorithm 8 in the case where $M = 1$. It is most likely that this is due to the choice of proposal distribution. The proposal for algorithms 7 is from "non-singleton to singleton" and from "singleton to non-singleton" therefore the transitions are likely to be "larger" when accepted.

The previously discussed inefficiency surrounding algorithms 1 and 6 is clearly visible in the autocorrelation times. After the burn-in period it appears as though the state of the Markov Chain reaches a configuration of parameters with high density and is therefore very slow to mix further. It can be seen in the results for algorithm 8 that the higher the value of M the better the autocorrelation times, and indeed are approaching those of algorithm 2, as expected.

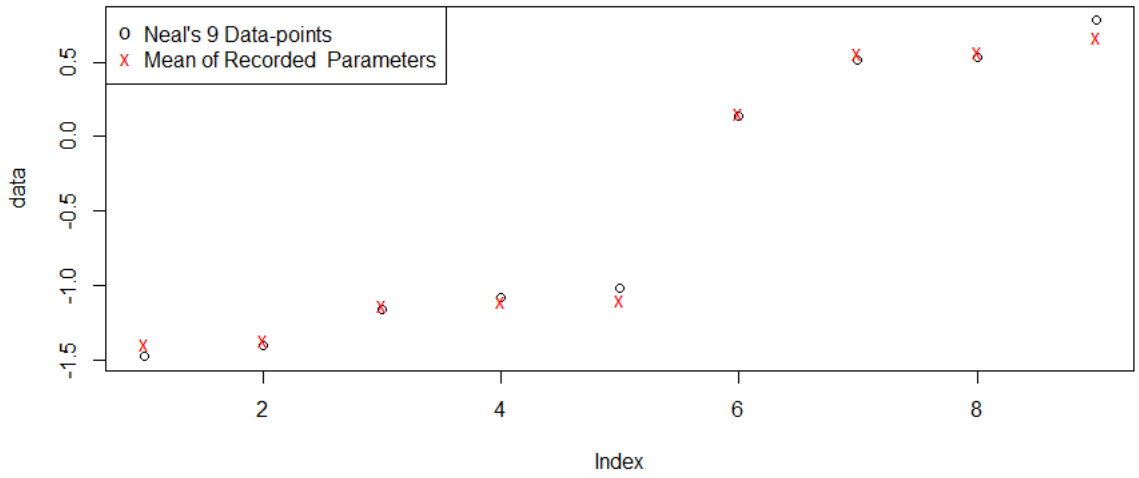


Figure 3: Algorithm 6: Mean of recorded $\{\theta_i\}_{i=1}^{9}$

The "x" points in figure (3) are the mean of the recorded $\{\theta_i\}$ values in the experiment detailed above. Denote $\theta_i^{(s)}$ as the θ_i recorded at the s^{th} iteration. Then the point with index i in the chart above are calculated as $\frac{1}{20{,}000}\sum_{s=1}^{20{,}000}\theta_i^{(s)}$. It can be seen that the average θ_i is very close to each of the data-points. This is a case of overfitting, where each data-point seems to belong to its own cluster for most iterations. Assuming that this is the case, one would expect the mean of the posterior normal distribution given the one data-point to be close to that data-point (the variance of the prior is low, and the mean is 0, hence the prior does not have a significantly large effect). Hence it appears that the algorithm is "valid" in the sense that:

$$\frac{1}{n}\sum_{s=1}^{n}\theta_i^{(s)} \xrightarrow[n\to\infty]{} \mathbb{E}_{\theta|y_i}[\theta_i]$$

6.2 2-Dimensional Normal Wishart Prior

Admittedly, 9 data-points may be quite a low number as it does not allow for a particularly wide range of partitions. The next step is to investigate higher dimensional data, with more data-points. The data was generated as follows:

$$x_{1:10} \sim N_2\left(\begin{pmatrix} 2 \\ -2 \end{pmatrix}, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right)$$

$$x_{11:20} \sim N_2\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right)$$

$$x_{21:30} \sim N_2\left(\begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right)$$

The DP mixture model used $\alpha = 1$ and base distribution G_0 , which is the Normal Inverse Wishart distribution:

$$NW\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, 2, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right)$$

This means: if $\theta \sim G_0$ then $\theta = (\mu, \Sigma)$

$$\mu \sim N_2\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \Sigma\right)$$

$$\Sigma^{-1} \sim \text{Wishart}\left(2, \begin{pmatrix} 1 & 0.5 \\ 0.5 & 1 \end{pmatrix}\right)$$

The assumed data-generating distribution is chosen as the multivariate normal:

$$F(\cdot, \theta) = N_2(\mu, \Sigma)$$

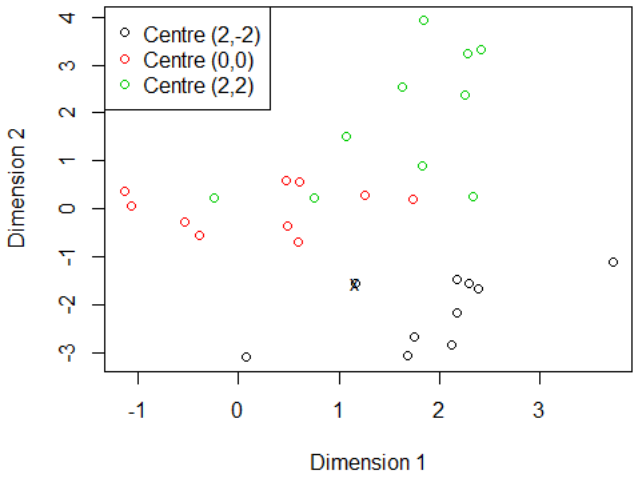


Figure 4: Multivariate Normal Data

The data-point corresponding to θ_1 is labelled in the plot above with an "x". The initial clustering is shown in the chart above by the colours, each algorithm was ran for 10,000 iterations after a burn-in period of 100 iterations. Again, the Gibbs sampling step corresponding to updating each of the component parameters given the data-points allocated to the component were carried out by taking advantage of the conjugacy and sampling from the posterior distributions.

	θ_1 μ_1	θ_1 μ_2	m
Algorithm 5 ($R = 4$)	2.000000	2.518588	2.771187
Algorithm 6 ($R = 4$)	14.208480	5.352208	14.60019
Algorithm 7	2.000000	1.840609	0.484223
Algorithm 8 ($M = 2$)	2.000000	1.914253	1.755413

Table 4: Autocorrelation Time

This parameter space, i.e. the support of the base distribution, is much larger as it is of a higher dimension than the Neal data and it is a Normal-Wishart distribution for both mean and variance rather than the normal prior for the mean. Unsurprisingly, algorithm 6 shows the same inefficiency discussed earlier. However, it appears the autocorrelation times for the parameters for the mean (μ) are very similar across the other algorithms. It is not immediately clear why this is the case. It may be due to lack of convergence, which has not been tested, or simply that the data-point (marked with an "x" in the figure above) is relatively close to $(0, 0)$ which is the mode of the prior distribution. In either case, there will be a higher probability of accepting a new component parameter than one of the existing parameters associated to other data-points. This may reduce the autocorrelation and explain why the algorithms have similar results. Encouragingly, the autocorrelation times for m, the number of unique components, vary for each algorithm. Similar to before, algorithm 8 performs better than algorithm 5, likely due to not having MH steps. However, the autocorrelation time for m for algorithm 7 is significantly lower than the others, and below 1. This means that the effective sample size from this algorithm is lower than one would expect from independent draws from the invariant distribution. This means that there is negative correlation in the number of components. The proposal distribution for algorithm 7 proposes transitions "singleton to non-singleton" and "non-singleton to singleton" therefore if the data was initialised to all singletons, and the acceptance probability is high for each MH step, then there may be some periodicity from high m to low m and then low m to high m after each iteration. This could explain the negative correlation.

7 Dirichlet Process Multinomial Logistic Model

7.1 Model Formulation

In addition to performing unsupervised clustering via component allocations, the Dirichlet Process Mixture Model can be applied to supervised classification. This is where the model is first fitted to a training set of data, which has class labels, then applied to a classify data points in a training set (without class labels).

The Dirichlet Process Multinomial Logistic model (DPMNL) was first discussed by [13], and is formulated as follows. Note that the notation is slightly different to that in previous sections in order to conform to the notation of [12], which can be used for a more complete treatment of the model.

$$\begin{aligned} G &\sim DP(\alpha, G_0) \\ \theta_i|G &\sim G \\ X_{i,j} &\sim N(\mu_{i,j}, \sigma^2_{i,j}) \end{aligned} \tag{23}$$

$$\mathbb{P}[Y_i = k|X_i, \theta_i] = \frac{\exp(a_{i,k} + \sum_{j=1}^{d} X_{i,j} b_{i,j,k})}{\sum_{k'=1}^{K} \exp(a_{i,k'} + \sum_{j=1}^{d} X_{i,j} b_{i,j,k'})} \tag{24}$$

$$i \in \{1, \ldots, n\}\ j \in \{1, \ldots, d\}\ k \in \{1, \ldots, K\}$$

$X_i = (X_{i,1}, \ldots, X_{i,d})$,$a_i = (a_{i,1}, \ldots, a_{i,K})$, $b_i = (b_{i,\cdot,1}, \ldots, b_{i,\cdot,K})$, and $b_{i,\cdot,k} = (b_{i,1,k}, \ldots, b_{i,d,k})$, $\mu_i = (\mu_{i,1}, \ldots, \mu_{i,d})$,$\sigma_i^2 = (\sigma^2_{i,1}, \ldots, \sigma^2_{i,d})$
$\theta_i = (a_i, b_i, \mu_i, \sigma_i^2)$
Note: the choice of (23) and (24) can be changed in a more general case, depending on the data at hand [12].
G_0 is some distribution such that for some i, a draw from G_0 would be $\mu_{i,}, \sigma_{i,}, a_{i,}, b_{i,}$. In particular [13] uses the following, whereby each realisation is drawn independently from the others and $m_\mu, m_\sigma, \sigma_\mu, \sigma_\sigma, \tau$,$\upsilon$ are known hyperparmaters (a development would be to attach a prior to these).

$$\mu_{i,j} \sim N(m_\mu, \sigma^2_\mu)$$

$$\log(\sigma^2_{i,j}) \sim N(m_\sigma, \sigma^2_\sigma)$$

$$a_{i,k} \sim N(0, \tau^2)$$

$$b_{i,j,k} \sim N(0, \upsilon^2)$$

The above model can be viewed as a DPMM by treating the covariates X_i and class label (response) Y_i as a single data point. The joint distribution of (X_i, Y_i) is $P(x_i, y_i) = P(y_i|x_i)P(x_i)$ where $P(y_i|x_i)$ and $P(x_i)$ are given in the formulation above by (24) and (23). Therefore, the likelihood function denoted "$f(\cdot|\theta_r)$" in previous sections is equivalent to:

$$f(x_i, y_i|\theta_r) = \left(\frac{\exp(a_{r,y_i} + \sum_{j=1}^{d} x_{i,j} b_{r,j,y_i})}{\sum_{k'=1}^{K} \exp(a_{r,k'} + \sum_{j=1}^{d} x_{i,j} b_{r,j,k'})} \right) \prod_{j=1}^{d} N(x_{i,j}|\mu_{r,,j}, \sigma^2_{r,,j}) \tag{25}$$

here $N(x_{i,j}|\mu_{r,j}, \sigma^2_{r,j})$ is the density of the normal distribution, $N(\mu_{r,j}, \sigma^2_{r,j})$, evaluated at $x_{i,j}$.

Using the Model

The DPMNL model is a specific case of the DPMM. Therefore, for a given training set, $\{(x_i, y_i)|i \in T\}$ for some index set T, the model can be fitted using an MCMC procedure as detailed in algorithms 1-8 in the previous section. Fitting the model means recording the values $\{\theta_i|i \in T\}$ for numerous

iterations, possibly after some burn-in period to allow for convergence of the MCMC procedure first. Denote the parameters recorded at iteration $s \in S$ as $\theta^{(s)}$. Using the alternate view of the Dirichlet Process Mixture Model, $\theta^{(s)}$ is equivalent to $(\phi^{*(s)}, Z^{(s)})$ which are the component parameters and component allocations at iteration s, where $\phi^*_{z_i} = \theta_i$. This second representation may also be used in the fitting procedure.

Classifying a new unlabelled data-point x' is performed by first allocating the data-point to a component/cluster at each iteration s, then recording the probability of being in each class given the cluster allocation. These probabilities are then averaged over the $s \in S$ iterations, to give K average probabilities (one for each class). The maximum of these is then chosen for the class prediction.

At each iteration, the $x' = (x'_1, \ldots, x'_d)$ is allocated to component with highest likelihood where the likelihood for cluster with parameter θ_i is calculated as $\prod_{j=1}^{d} N(x'_j | \mu_{i,j}, \sigma^2_{i,j})$. The probability of being in class k at this iteration is $\mathbb{P}[Y_i = k | x'_i, \phi_z^{*(s)}](24)$ where, $\phi_z^{*(s)}$ is the component parameter of the cluster allocation, z, given to x' at iteration s. The average class allocation probability for each class $k \in \{1, \ldots K\}$ are given by:

$$\frac{1}{|S|} \sum_{s=1}^{|S|} \mathbb{P}[Y_i = k | x'_i, \theta_z^{(s)}]$$

Data-point x' is therefore labelled as the class with the highest average class allocation probability.

Discussion

The intuition behind this model is that during the "fitting" procedure, where both the covariates and class label are observed, the covariate-label pair will be jointly clustered and allocated a parameter set at each iteration. Therefore, the covariates are clustered in such a way that they have a similar relationship to the class label. The parameters for a multinomial logistic model are allocated for each cluster, where the clustering is carried out in such a way that the logistic model for that cluster has some predictive power. When the labels are not observed, i.e. in the test case, the covariates are probabilistically clustered into the clusters identified by the training cases which are similar based on the likelihood given by (23). It is hoped that if the test covariates are similar to the training covariates in the allocated cluster, then the relationship to the class label will also be similar.

7.2 Experimentation

The model shall be verified as "working" by applying it to artificially generated data, then observing the classification performance. The model shall then be applied to a real data-set and the classification performance will be compared to that of other classifiers.

Artificial Data

The data in question is 300 multinomial data-points generated in the same way as in section 6.2. Where the class labels correspond to the 3 multinomial distributions, 100 data-points are drawn for each class. The data was split into 200 training cases and 100 test cases. The data is be summarised below, where each colour corresponds to a class and the "x" marks correctly classified test data-points.

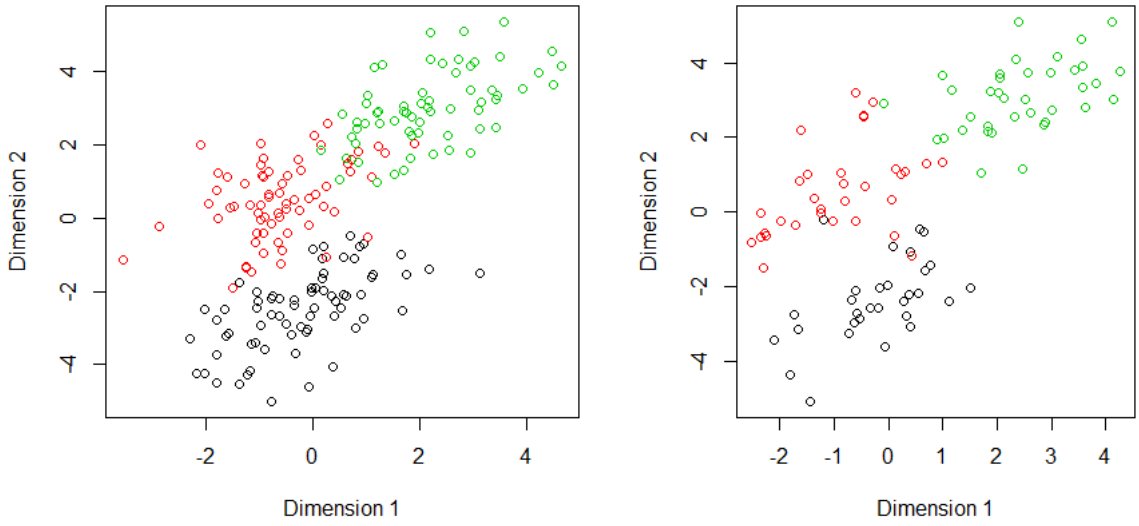


Figure 5: Training Data (left) and Test data (right)

The model used was as described above with hyperparameters:

$$m_\mu = 0,\ m_\sigma = 0,\ \sigma_\mu = 1,\ \sigma_\sigma = 2,\ \tau = 1,\ \upsilon = 1$$

The MCMC procedure for the the training data was carried out using Algorithm 8 with $m = 3$, $\alpha = 0.5$. The updating of parameters was carried out using a Metropolis Hastings step repeated 5 times for each iteration, where the proposal distribution was the prior for each of $a_k, b_k, \mu_k, \sigma_k^2$, where k denotes the component allocation. The kernel corresponding to updating the parameter is invariant, aperiodic and irreducible, therefore, composing a cycle of 5 will again be invariant, aperiodic and irreducible by the cycle property. As the proposal is the prior, the acceptance probability is the ratio of likelihoods.

If the proposal parameters are $\phi_k^{*'} = (a'_k, b'_k, \mu'_k, \sigma_k^{2'})$ and the existing state has parameters $\phi_k^* = (a_k, b_k, \mu_k, \sigma_k^2)$ then the Metropolis Hastings step is split into two steps corresponding to updating (μ_k, σ_k^2) and then (a_k, b_k) .
The acceptance probability for the first is:

$$a\left((\mu_k, \sigma_k^2), (\mu'_k, \sigma_k^{2'})\right) = \min\left\{1, \prod_{i|z_i=k}\left(\frac{\prod_{j=1}^d N(x_{i,j}|\mu'_{i,j}, \sigma_{k,j}^{2'})}{\prod_{j=1}^d N(x_{i,j}|\mu_{i,j}, \sigma_{k,j}^2)}\right)\right\}$$

The second is:

$$a((a_k, b_k), (a'_k, b'_k)) = \min\left\{1, \prod_{i|z_i=k}\left(\frac{\frac{\exp(a'_{k,y_i} + \sum_{j=1}^d x_{i,j} b'_{k,j,y_i})}{\sum_{r=1}^K \exp(a'_{k,r} + \sum_{j=1}^d x_{i,j} b'_{k,j,r})}}{\frac{\exp(a_{k,y_i} + \sum_{j=1}^d x_{i,j} b_{k,j,y_i})}{\sum_{r=1}^K \exp(a_{k,r} + \sum_{j=1}^d x_{i,j} b_{k,j,r})}}\right)\right\}$$

The MH step is guaranteed to have the correct invariant distribution, therefore, is a valid replacement in step 2 for algorithm 8.

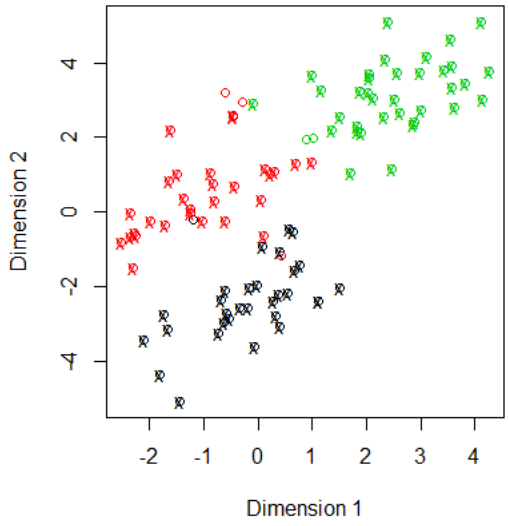


Figure 6: Correctly Classified Data-Points

The plot above shows the data-points correctly classified marked with both "o" and "x". This artificial data-set has a relatively large degree of separation between the covariates for each class, therefore it is unsurprising that the model performs well, with 94% accuracy.

7.2.1 Wine Data

The next task is to perform supervised clustering on a real-world data-set of higher dimension. An interesting data-set is The Wine Data-set [14] from the University of California, collected in 1991. This data consists of 13 attributes, and 3 classes. To make computation manageable, 4 attributes of the 13 were selected (Alcohol, Magnesium, Flavanoids, Color intensity) and the 3 classes are regions in Italy where the wine was sourced from. These attributes were first standardised by subtracting the mean and dividing by the standard deviation. The data was split randomly into 60 cases of training data-points and 118 instances of test data-points. The reason for this was the time it takes to fit the DPMNL model (100 iterations took 15 minutes). The same model that was used for the artificial data (above) was used but with the following hyperparameters:

$$m_\mu = 0,\ m_\sigma = 0,\ \sigma_\mu = 1,\ \sigma_\sigma = 1,\ \tau = 1,\ \upsilon = 1$$

Again algorithm 8 was used with $\alpha = 1$, $m = 3$, with the same Metropolis Hastings update to the component parameters.
A quick exploration of the data is shown below. The cross-sectional plots in 2-dimension suggests a degree of separation so it should be possible to classify the data with a relatively high accuracy. The three colours correspond to the three classes.

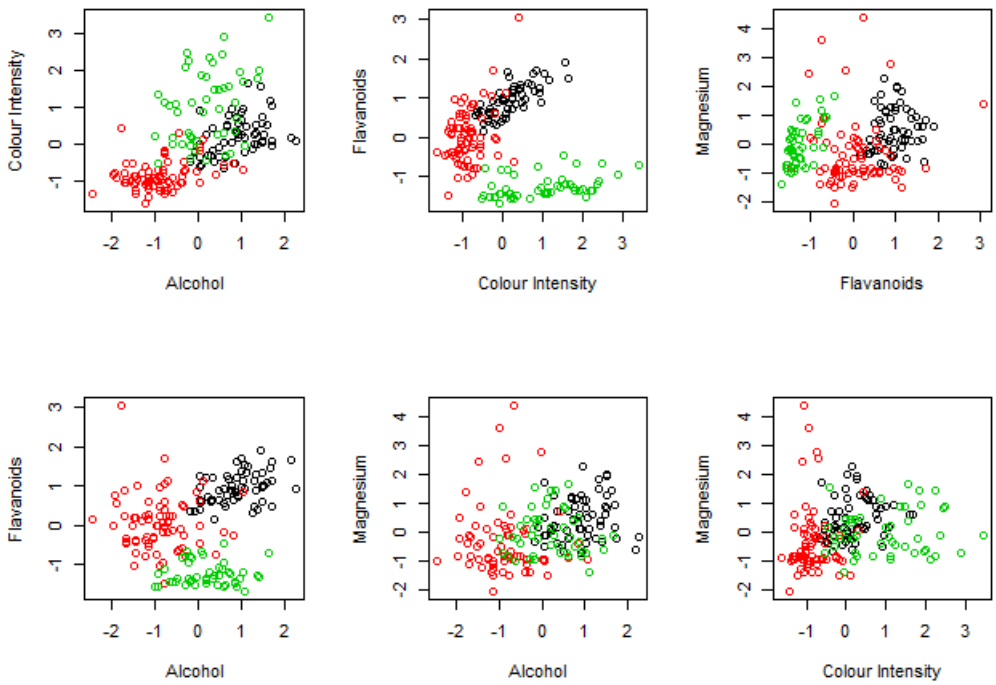


Figure 7: Wine data

Accuracy [20] gives the total percentage of correctly classified points, macro precision gives the average percentage of positively classified data-points that are correctly classified across classes. Macro recall gives the average percentage of truly positive data-points that are correctly classified across classes. Therefore, informally, precision is a measure of trust in the positively classified points after classification , recall is a measure of coverage. For example, a classifier that only classifies 1 point of 1000 as positive, and it is positive, would have the maximum precision of 1, but a recall of only 0.001 if all 1000 points are truly positive. The performance is compared to other classifiers. The SVM (support vector machine) and Naive Bayes were implemented using the R CRAN package "e1071" [31]with default parameters, and the Random Forest classification was implemented using the "randomForest" CRAN package [32].

A thorough investigation would involve some sort of cross-validation. However, as it is very time-consuming to fit the model, this has been overlooked. The single case should be sufficient to give an insight into how the DPMNL can be used and how it compares to other classifiers. A very brief performance summary against other methods is shown below. It is clear from the table below that the random forest procedure performs a superior supervised classification, having higher accuracy, precision and recall, followed by SVM (radial), the DPMNL model, SVM (linear) then finally Naive Bayes. The table below shows that the DPMNL is a reasonable classifier compared to other very common classifiers, outperforming Naive Bayes and SVM (linear), and being very similar to SVM (radial). In addition, given that it is a probabilistic model it has other advantages such as being able to handle missing values, though also has a large disadvantage of being time consuming to fit. The DPMNL can also be extended using priors for the hyperparameters which has been shown to improve the performance significantly by [13] and other Generalized Linear Models may be used instead of the Multinomial Logistic to improve the performance, as explained in [12].

	Accuracy	Macro Precision	Macro Recall
Naive Bayes	0.9152	0.9103	0.9276
SVM (linear)	0.9237	0.9186	0.9415
DPMNL	0.9406	0.9352	0.9331
SVM (Radial)	0.9491	0.94028	0.9604
Random Forest	0.9576	0.9444	0.9685

Table 5: Performance Measures

8 Hierarchical Dirichlet Process Mixture Model

The motivation for the Hierarchical DP (HDP) Mixture Model is primarily in its ability to model data which is separated into groups, where the groups are not independent. The dependency structure is given by figure (8), where the data is split into M groups, indexed by j. The plate diagram notation can be understood using the explanation given in section 3.

Using the notation of figure (8), the HDP is a collection DP mixture models indexed by j, whereby each G_j has the base distribution G_0, which is again a Dirichlet Process with base distribution H. The groups of data are linked through their joint dependency on G_0. Further intuition is that the first DP, G_0, is a discretization of base distribution H, i.e. a discrete distribution over the support of H. Informally, this discrete distribution, G_0, is a "high level" distribution of component parameters $\{\phi_k^*\}_k$ using the data across groups, as though the data was not grouped. The lower level DPs, $\{G_j\}_j$, are discretizations of G_0, so essentially a re-weighted discrete distribution over the support of G_0 specific to each group of data. The weights associated to G_j correspond only to the data in group j.

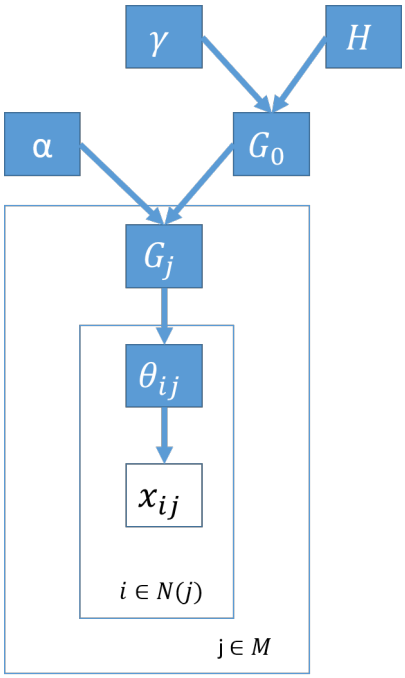


Figure 8: Plate Diagram: Hierarchical Dirichlet Process Mixture Model

The HDP model can be written in full for $j \in M$, $i \in N(j)$, as:

$$G_0|\gamma, H \sim DP(\gamma, H)$$

$$G_j|\alpha, G_0 \sim DP(\alpha, G_0)$$

$$\theta_{i,j}|G_j \sim G_j$$

$$x_{i,j} \sim F(\cdot, \theta_{i,j})$$

It is known from the stick breaking construction (5) that:

$$G_0 = \sum_{k=1}^{\infty} \pi_k^{(0)} \delta_{\phi_k}$$

where $\forall k \in \mathbb{N}, \pi^{(0)} = \{\pi_k^{(0)}\}_{k=1}^{\infty} \sim GEM(\gamma)$, $\phi_k \sim H$ independently from each other. Note that the GEM distribution is described in the earlier section on Dirichlet Processes.

$$G_j = \sum_{k=1}^{\infty} \pi_{j,k} \delta_{\phi_k}$$

where $\pi_j = \{\pi_{j,k}\}_{k=1}^{\infty}$.
The dependency structure is given in the plate diagram, but to be explicit the vectors π_j are independent given $\pi^{(0)}$ as G_j are each conditionally independent given G_0.
The following illustrates an interesting property of the HDP, the explanation has been modified from that given in [6]. To keep notation consistent, let A be the support of H, and let $(A_1, \ldots, A_r)$ be any measurable partition of A and $K_l = \{k | \phi_k^* \in A_l\} \forall l \in \{1, \ldots, r\}$. Assuming H is non-atomic, which is the case for the application explained later, the ϕ_k^* values are distinct with probability one. The $\{K_1, \ldots, K_r\}$ then form a partition over the positive integers corresponding to the partition over $(A_1, \ldots, A_r)$ over A. Therefore, using Ferguson's definition of the DP:

$$(G_j(A_1), \ldots, G_j(A_r)) \sim \text{Dirichlet}(\alpha G_0(A_1), \ldots, \alpha G_0(A_r))$$

As $G_j(A_i) = \sum_{k \in K_i} \pi_{j,k}$ and $G_0(A_i) = \sum_{k \in K_i} \pi_k^{(0)}$:

$$(\sum_{k \in K_1} \pi_{j,k}, \ldots, \sum_{k \in K_r} \pi_{j,k}) \sim \text{Dirichlet}(\alpha \sum_{k \in K_1} \pi_k^{(0)}, \ldots, \alpha \sum_{k \in K_r} \pi_k^{(0)})$$

Therefore for any partition $\{K_1, \ldots, K_r\}$:

$$\pi_j \sim DP(\alpha, \pi^{(0)})$$

Let $z_{i,j}$ be allocation variables for each data-point $x_{i,j}$ such that $\theta_{i,j} = \phi^*_{z_{i,j}}$ where the $x_{i,j}$ and $\theta_{i,j}$ correspond to the representation given in the diagram above without allocation variables.

$$\pi^{(0)} | \gamma \sim GEM(\gamma) \qquad z_{i,j} \sim \pi_j$$
$$\pi_j | \alpha, \pi^{(0)} \sim DP(\alpha, \pi^{(0)}) \qquad x_{i,j} | z_{i,j}, \{\phi_k^*\}_{k=1}^{\infty} \sim F(\cdot, \phi^*_{z_{i,j}})$$
$$\phi_k^* \sim H$$

where the component distribution F is a modeling choice, as it was with the regular Dirichlet Process Mixture Model.

8.1 Chinese Restaurant Franchise

Just as with the Dirichlet Process there is the Chinese Restaurant Process (CRP) analogy, there is the Chinese Restaurant Franchise (CRF) analogy for the HDP, explained again by [6]. For convenience, the notation has been kept the same. Let there be a chain of M restaurants, and numerous tables at each restaurant. There is a set menu shared across the franchise, the menu has dishes $\{\phi_k^*\}_k$. Each restaurant is identified by index $j \in \{1, \ldots M\}$. Let the dish of customer i in restaurant j be denoted $\theta_{i,j}$. Let the table of customer i at restaurant j be given by $t_{i,j}$. It is assumed each table has the same dish.

The first customer to arrive at each table orders a dish, and it is shared amongst all customers sitting at this table. The "dish" can be viewed as a banquet meal if it makes the metaphor easier. Therefore $\theta_{i,j} = \theta_{i',j}$ if $t_{i,j} = t_{i',j}$. Using the notation of [6], let the dish allocated to table t in restaurant j be denoted $\psi_{j,t}$ and the dish allocation be denoted $k_{j,t}$ whereby $\phi^*_{k_{j,t}} = \psi_{j,t}$.

Customers are grouped into tables within each restaurant, and customers across restaurants may

be grouped according to the dishes. Note that multiple tables in multiple restaurants may be served the same dish, this is due to the discreteness of the Dirichlet Process in the base distribution, G_0.

Notation-wise, let $n_{j,t,k}$ be the number of customers in restaurant j at table t being served dish k. Use dot notation for marginals, therefore $n_{j,\cdot,k}$ denotes the number of customers in restaurant j with dish k. $n_{j,t,\cdot}$ denotes the number of customers in restaurant j at table t. For table counts, let $m_{j,k}$ be the number of tables in restaurant j serving dish k, $m_{\cdot,k}$ denotes the number of tables in all restaurants serving dish k, and let $m_{j,\cdot}$ be the number of tables in restaurant j.

Given an initial state of the franchise, a new customer indexed i, entering restaurant j would be served dish $\theta_{i,j}$ according to the following distribution:

$$\theta_{i,j}|\theta_{1:i-1,j},\alpha,G_0 \sim \sum_{t=1}^{m_{j,\cdot}} \frac{n_{j,t,\cdot}}{i-1+\alpha}\delta_{\psi_{j,t}} + \frac{\alpha}{i-1+\alpha}G_0 \tag{26}$$

This is exactly the same as seen with the Chinese Restaurant Process, when G_0 is known. If a term $\psi_{j,t}$ for $t \leq m_{j,\cdot}$ is drawn then set $\theta_{i,j} = \psi_{j,t}$, otherwise increase $m_{j,\cdot}$ by 1, set $t_{i,j}$ as this newly increased $m_{j,\cdot}$ and draw a new $\psi_{j,t_{i,j}}$ from G_0.

G_0 may be marginalised as it is not observed. Let $D(j) = \{\psi_{j,t}|\forall t\}$, i.e all the dishes from restaurant j. Given G_0, $\theta_{i,j}$is independent of $D(j')$ $\forall j' \neq j$. However, G_0 is dependent on $D(j')$ $\forall j'$.G_0 is a DP from which the values in the set $\cup_{j'} D(j')$,i.e. $\{\psi_{1,1},\ldots,\psi_{2,1},\ldots,\ldots\}$, have all been drawn. Let $D_{-j,t}$ be the set of all the previous values drawn excluding $\psi_{j,t}$. Again, using the predictive distribution of a DP:

$$\psi_{j,t}|D_{-j,t},\gamma,H \sim \sum_{k=1}^{K} \frac{m_{\cdot,k}}{m_{\cdot,\cdot}+\gamma}\delta_{\phi_k^*} + \frac{\gamma}{m_{\cdot,\cdot}+\gamma}H \tag{27}$$

where K is the number of unique values in $D_{-j,t}$, H is the base distribution of G_0, and the $m_{\cdot,k}$ and $m_{\cdot,\cdot}$are as defined previously but for the state of the franchise excluding table j,t.

Let $\boldsymbol{S}$ denote all table and dish allocations in addition to all parameters associated to each dish allocation. The equations above are used to sample for $\theta_{i,j}$, given the state of the franchise, $\boldsymbol{S}$, with unknown G_0, known α, H and γ.This is done by sampling from the distribution:

$$\theta_{i,j}|\boldsymbol{S},\alpha,\gamma,H \sim \sum_{t=1}^{m_{j,\cdot}} \frac{n_{j,t,\cdot}}{i-1+\alpha}\delta_{\psi_{j,t}} + \frac{\alpha}{i-1+\alpha}Q$$

where Q is the distribution given in (27).

Posterior Sampling using the Chinese Restaurant Franchise

Let $\boldsymbol{k} = \{k_{j,1:m_{j,\cdot}}\}_{j=1}^{M} = \{k_{j,t}|t \in \{1,\ldots,m_{j,\cdot}\}, j \in \{1,\ldots,M\}\}$, $\boldsymbol{t} = \{t_{i,j}|i \in \{1,\ldots,n_{j,\cdot,\cdot}\}, j \in \{1,\ldots,M\}\}$. Let $\boldsymbol{k}_{-j,t}$ be $\boldsymbol{k}$ excluding table t at restaurant j, and $\boldsymbol{t}_{-i,j}$ be $\boldsymbol{t}$ excluding data-point i,j. $t_{(-i),j} = \{t_{i',j}|i' \in \{1,\ldots,n_{j,\cdot,\cdot}\}\backslash\{i\}\}$. In addition, the corresponding counts excluding data-point i,j and table j,t are $n_{j,t,k}^{-i,j}$ and $m_{j,k}^{-j,t}$. The number of unique component parameters is K. The corresponding conditional distributions in the Chinese Restaurant Franchise are:

$$\mathbb{P}[t_{i,j} = t'|\boldsymbol{t}_{-i,j},\boldsymbol{k}] = \sum_{t=1}^{m_{j,\cdot}} \frac{n_{j,t,\cdot}^{-i,j}}{n_{j,\cdot}-1+\alpha}\delta_t(t') + \frac{\alpha}{n_{j,\cdot}-1+\alpha}\delta_{t_{new}}(t')$$

$$\mathbb{P}[k_{j,t} = k'|\boldsymbol{t}_{-i,j},\boldsymbol{k}] = \sum_{k=1}^{K\cdot} \frac{m_{\cdot,k}^{-j,t}}{m_{\cdot,\cdot}-1+\gamma}\delta_k(k') + \frac{\gamma}{m_{\cdot,\cdot}-1+\gamma}\delta_{k_{new}}(k')$$

Hence by Bayes' theorem, the posteriors are:

$$\mathbb{P}[t_{i,j} = t' | \boldsymbol{t}_{-i,j}, \boldsymbol{k}] \propto \sum_{t=1}^{m_{j,\cdot}} \frac{n_{j,t,\cdot}^{-i,j} f(x_{i,j}|\phi^*_{k_{j,t}})}{n_{j,\cdot} - 1 + \alpha} \delta_t(t') + \frac{\alpha q(x_{i,j})}{n_{j,\cdot} - 1 + \alpha} \delta_{t_{new}}(t')$$

where $q(x_{i,j}) = \sum_{k=1}^{K_\cdot} \frac{m_{\cdot,k}^{-j,t}}{m_{\cdot,\cdot} - 1 + \gamma} + \frac{\gamma}{m_{\cdot,\cdot} - 1 + \gamma} \int f(x_{i,j}|\phi^*) dH(\phi)$
Changing the component allocation for each table is dependent on all data-points in that table. The distribution for the component allocation of table j,

$$\mathbb{P}[k_{j,t} = k' | \boldsymbol{t}_{-i,j}, \boldsymbol{k}] \propto \sum_{k=1}^{K_\cdot} \frac{m_{\cdot,k}^{-j,t} F_{j,t}(k)}{m_{\cdot,\cdot} - 1 + \gamma} \delta_k(k') + \frac{\gamma F_{j,t}(k_{new})}{m_{\cdot,\cdot} - 1 + \gamma} \delta_{k_{new}}(k')$$

where $F_{j,t}(k) = \prod_{i',j'|t'_{i',j'}=t,j'=j} f(x_{i',j'}|\phi^*_k)$ and $F_{j,t}(k_{new}) = \int \prod_{i',j'|t'_{i',j'}=t,j'=j} f(x_{i',j'}|\phi^*_k) dH(\phi)$ which are the likelihoods of every data-point allocated to table j,t being allocated to each component allocation k or k_{new}.

CRF Algorithm for Posterior Sampling

At each iteration:

1. For each $j \in \{1, \ldots, J\}$:
   (a) For each $i \in \{1, \ldots, n_{j,\cdot}\}$:
      i. Draw $t_{i,j}$ from:

$$\mathbb{P}[t_{i,j} = t' | \boldsymbol{t}_{-i,j}, \boldsymbol{k}] \propto \sum_{t=1}^{m_j^{-i,j},\cdot} \frac{n_{j,t,\cdot}^{-i,j} f(x_{i,j}|\phi^*_{k_{j,t}})}{n_{j,\cdot} - 1 + \alpha} \delta_t(t') + \frac{\alpha q(x_{i,j})}{n_{j,\cdot} - 1 + \alpha} \delta_{t_{new}}(t') \tag{28}$$

      ii. If the newly drawn $t_{i,j}$ is not an existing allocation draw a $k_{j,t_{i,j}}$using the following probabilities:

$$\mathbb{P}[k_{j,t} = k' | \boldsymbol{t}_{-i,j}, \boldsymbol{k}] \propto \sum_{k=1}^{K^{-j,t}\cdot} \frac{m_{\cdot,k}^{-j,t} f(x_{i,j}|\phi^*_k)}{m_{\cdot,\cdot} - 1 + \gamma} \delta_k(k') + \frac{\gamma F_{-j,t}(k_{new})}{m_{\cdot,\cdot} - 1 + \gamma} \delta_{k_{new}}(k') \tag{29}$$

      iii. If $k_{j,t_{i,j}}$ is a new component, draw a parameter from:

$$\phi^*_{k_{new}} \sim H | x_{i,j} \tag{30}$$

   (b) For each $t \in \{1, \ldots, m_{j,\cdot}\}$
      i. Sample $k_{j,t}$from:

$$\mathbb{P}[k_{j,t} = k' | \boldsymbol{t}_{-i,j}, \boldsymbol{k}] \propto \sum_{k=1}^{K^{-j,t}\cdot} \frac{m_{\cdot,k}^{-j,t} F_{-j,t}(k)}{m_{\cdot,\cdot} - 1 + \gamma} \delta_k(k') + \frac{\gamma F_{-j,t}(k_{new})}{m_{\cdot,\cdot} - 1 + \gamma} \delta_{k_{new}}(k') \tag{31}$$

      ii. If $k_{j,t}$ is a new component, draw a parameter from:

$$\phi^*_{k_{new}} \sim H | \{x_{i,j} \text{ such that } t_{i,j} = t\} \tag{32}$$

2. For each $k \in \{1, \ldots K\}$
   (a) Draw ϕ^*_k from H given all data-points across all J groups allocated to component k.

$$\phi^*_k \sim H | x_{i,j} \text{ such that } k_{j,t_{i,j}} = k \tag{33}$$

Stationarity, Irreducibility and Aperiodicity

The state in the Markov Chain is $(\boldsymbol{t}, \boldsymbol{k}, \phi^*_{1:K})$, given some data $\{x_{i,j}\}_{i,j}$, the probability of the state is given by the following density:

$$P(\boldsymbol{t}, \boldsymbol{k}, \phi^*_{1:K} | \{x_{i,j}\}_{i,j}) \propto \prod_{i,j} f(x_{i,j} | \phi^*_{k_{j,t_{i,j}}}) P(\boldsymbol{t}, \boldsymbol{k}) \prod_{k=1}^{K} h(\phi^*_k)$$

By exchangeability, one may write:

$$P(\boldsymbol{t}, \boldsymbol{k}) = P(t_{i,j} | \boldsymbol{t}_{-t,j}, \boldsymbol{k}) P(\boldsymbol{t}_{-i,j}, \boldsymbol{k}) = P(k_{j,t} | \boldsymbol{k}_{-j,t}, \boldsymbol{t}) P(\boldsymbol{k}_{-j,t}, \boldsymbol{t}) = P(t_{i,j}, k_{j,t} | \boldsymbol{t}_{-i,j}, \boldsymbol{k}_{-j,t}) P(\boldsymbol{t}_{-i,j}, \boldsymbol{k}_{-j,t})$$

The kernel associated with the Chinese Restaurant Franchise algorithm is a cycle of kernels corresponding to steps: 28, 29, 31, and 33. This is Gibbs sampling using the posterior distributions given above. Similar to algorithm 2, it is difficult to see directly that the algorithm above uses Gibbs sampling. Instead, it is easier to see that step 1 is a translation of Gibbs sampling using the urn-representation described by equations 26 and 27 into the CRF representation. It can also be seen that the Markov Chain used in the algorithm has the correct invariant distribution directly using detailed balance. This can be done by looking at the three cases depending on $t_{i,j}$ and $k_{j,t_{ij}}$. Either $t_{i,j}$ and $k_{j,t_{ij}}$ are both singletons; $t_{i,j}$ is a non-singleton or the final case is that $t_{i,j}$ is a singleton and $k_{j,t_{ij}}$ is a non-singleton allocation. Let the kernel corresponding to step (1a) be denoted $K^{(i,j)}_{CRF}\left((t_{i,j}, k_{j,t_{ij}}), (t'_{i,j}, k'_{j,t_{ij}})\right)$. In addition, in a single sweep there is positive probability of transitioning from any $(t_{i,j}, k_{j,t_{ij}})$, to any other $(t'_{i,j}, k'_{j,t_{ij}})$. Hence the cycle of kernels (i.e. the updates from kernel $K^{(i,j)}_{CRF}$ for all i, j) is both irreducible and aperiodic. It can be seen that the element-wise kernel updates the $t_{i,j}$ variable, then if a "new" allocation is sampled, the next step is to update the $k_{j,t_{i,j}}$ variable. This second update precisely corresponds to kernel for step (1b). Therefore, using a similar justification as above, the kernel corresponding to this step satisfies detailed balance, irreducibility and aperiodicity. Given that the final update of component parameters is justified in the same way as that for algorithm 2, this algorithm is valid.

8.2 Topic Modelling

An application of the Hierarchical Dirichlet Process Mixture Model is in topic modelling across multiple documents, using an an extension of the Latent Dirichlet Allocation (LDA) method. Let there be M documents, which are treated as 'bags of words', with words in no particular order, each document is indexed by $j \in \{1, \ldots, M\}$. The total collection of unique words is called the corpus, and the number of unique words across all the documents is denoted V. The Hierarchical LDA is a Hierarchical Mixture Model whereby the base distribution denoted H in figure 8 is a V- dimensional Dirichlet distribution, and each realised draw from H, denoted θ (or ϕ^*) is a V-dimensional vector that can be interpretted as a categorical probability distribution over words, this is called a topic. Document j is associated with a Dirichlet Process G_j with base distribution G_0 which is also a DP with base distribution H. All $\{G_j\}_{j=1}^M$ share the same base distribution G_0. The discrete property of the Dirichlet Process G_0 allows for multiple topics to be shared across documents, and the discreteness of each DP G_j means that multiple words in the same document have positive probability of being associated with the same topic. In the HDP notation H is a V-dimensional Dirichlet distribution, $\{\phi^*_k\}_k$,$\{\theta_{i,j}\}_{i,j}$ are V-dimensional vectors whose elements sum to 1, $F(\cdot, \theta_{i,j})$ is a categorical distribution with parametrised by $\theta_{i,j}$.

The Hierarchical DP Topic Model will group words into "tables" in each document, then associate each "table" to a component, which corresponds to a topic. This allows the same word to be associated to multiple topics within each document as well as across documents. For example, in the results below, the words "putin" and "trump" are associated to multiple topics.

Given documents may be rather large, it is possible that the corpus may contain thousands of words. The model proposed however is a conjugate model, therefore it is possible to extend algorithm 3, where the parameters are marginalised out, to the HDP to perform the MCMC algorithm more efficiently.The validity of this marginalisation is justified by the proof in section 4.3 where the kernel in the CRF algorithm has been reduced by integrating out the component parameters. This marginalised algorithm is detailed in [6], has state $(\boldsymbol{t}, \boldsymbol{k})$ and is implemented as follows:

Let the marginal likelihoods be:

$$F *_{x_{i,j}} (k) = \frac{\int f(x_{i,j}|\phi) \prod_{i',j' \neq i,j | k_{j',t_{i',j'}}=k} f(x_{i',j'}|\phi) dH(\phi)}{\int \prod_{i',j' \neq i,j | k_{j',t_{i',j'}}=k} f(x_{i',j'}|\phi) dH(\phi)}$$

$$F *_t (k) = \frac{\int \prod_{i,j|t_{i,j}=t} f(x_{i,j}|\phi) \prod_{i',j'|t_{i',j'} \neq t, k_{j',t_{i',j'}}=k} f(x_{i',j'}|\phi) dH(\phi)}{\int \prod_{i',j'|t_{i',j'} \neq t, k_{j',t_{i',j'}}=k} f(x_{i',j'}|\phi) dH(\phi)}$$

where H is the Dirichlet distribution of dimension V, and some concentration parameter α. $f(x_{i,j}|\phi) = p_i$ where $\phi = (p_1, \ldots, p_V)$. The above integrals, and integrals such as $\int \prod_{i,j|t_{i,j}=t} f(x_{i,j}|\phi) dH(\phi)$ can be computed easily using equation (1).

Marginalised CRF Algorithm for Posterior Sampling

At each iteration:

1. For each $j \in \{1, \ldots, J\}$:
   (a) For each $i \in \{1, \ldots, n_{j,\cdot}\}$:
      i. Draw $t_{i,j}$ from:

$$\mathbb{P}[t_{i,j} = t' | \boldsymbol{t}_{-i,j}, \boldsymbol{k}] \propto \sum_{t=1}^{m_j^{-i,j}{}_{,\cdot}} \frac{n_{j,t,\cdot}^{-i,j} F *_{x_{i,j}} (k_{j,t})}{n_{j,\cdot} - 1 + \alpha} \delta_t(t') + \frac{\alpha q(x_{i,j})}{n_{j,\cdot} - 1 + \alpha} \delta_{t_{new}}(t') \tag{34}$$

      where $q(x_{i,j}) = \sum_{k=1}^{K_\cdot} \frac{m_{\cdot,k}^{-j,t}}{m_{\cdot,\cdot} - 1 + \gamma} + \frac{\gamma}{m_{\cdot,\cdot} - 1 + \gamma} \int f(x_{i,j}|\phi^*) dH(\phi)$

      ii. If the newly drawn $t_{i,j}$ is not an existing allocation draw a $k_{j,t_{i,j}}$using the following probabilities:

$$\mathbb{P}[k_{j,t} = k' | \boldsymbol{t}_{-i,j}, \boldsymbol{k}] \propto \sum_{k=1}^{K^{-j,t}_\cdot} \frac{m_{\cdot,k}^{-j,t} F *_{x_{i,j}} (k)}{m_{\cdot,\cdot} - 1 + \gamma} \delta_k(k') + \frac{\gamma \int f(x_{i,j}|\phi) dH(\phi)}{m_{\cdot,\cdot} - 1 + \gamma} \delta_{k_{new}}(k') \tag{35}$$

   (b) For each $t \in \{1, \ldots, m_{j,\cdot}\}$
      i. Sample $k_{j,t}$from:

$$\mathbb{P}[k_{j,t} = k' | \boldsymbol{t}_{-j,t}, \boldsymbol{k}] \propto \sum_{k=1}^{K^{-j,t}_\cdot} \frac{m_{\cdot,k}^{-j,t} F *_t (k)}{m_{\cdot,\cdot} - 1 + \gamma} \delta_k(k') + \frac{\gamma \int \prod_{i,j|t_{i,j}=t} f(x_{i,j}|\phi) dH(\phi)}{m_{\cdot,\cdot} - 1 + \gamma} \delta_{k_{new}}(k') \tag{36}$$

8.2.1 Tweet Data

It is difficult to perform a rigorous analysis of whether the topic modelling "works" and produces reasonable clusters as the clusters do not have labels. However, instead the clustering shall be performed on an interesting data-set of tweets relating to news-articles. The topics given by the model components will hopefully correspond to the news "topics" or "stories" to some degree. The words associated to each component will be displayed and the cluster quality will be given a very informal, and quite subjective, visual assessment.

A detailed description of how to extract tweets and process the Twitter data using R is given by [15] and uses the following packages: [21, 22, 23, 24, 25]. 800 tweets[38] were downloaded from various news outlets. These news outlets have been annonymised in accordance with the Twitter API policy.

To reduce the data into clear themes, only tweets including words "trump", "corbyn", "china", "sex", "europe", "migrant", "putin", "cameron", "sanders", "eu" were included as these were in the top 20 most frequent words, and were chosen rather subjectively as being interesting. This left 62 tweets shown in figure (10) in the appendix, each treated as individual documents in the Topic Model framework, with $V = 335$ unique words in total.

The model was initiated with all the words in each document being allocated to a single "table" , and each "table" in each document being allocated to a component unique to that document, i.e. 62 components. The hyperparameters were set as $\alpha = \gamma = 0.5$.

Results

The implementation shown in the R-code section was very slow, and therefore only 100 iterations were carried out. It is possible that more meaningful results may be obtained by some sort of averaging procedure over the numerical values of the state in the Markov Chain across iterations, however, in the interest of simplicity, the state from 100th iteration alone shall be looked at. The motivation for this is the hope that the 100th iteration will be a "representative" sample of the state, meaning that it is hoped that this state sampled has relatively high probability over all configurations of component and "table" allocations.

In each document, the words were grouped into "tables" and then the "tables" were associated to components. 11 components were "learnt" from the data. The breakdown of the number of words associated to each component can be seen below. A document is associated to a component if it contains a word or "table" associated to the component. Given that there are 62 documents and 89 component associations, it is clear that some documents (tweets) have been associated to multiple components. This is as one would hope from the HDP.

Component	1	2	3	4	5	6	7	8	9	10	11	Total
"tables" Count	10	9	30	10	9	8	4	9	4	4	9	106
Words Count	38	45	151	48	45	61	22	45	27	28	55	565
Document Count	8	7	24	8	8	6	4	7	4	4	9	89

Table 6: Summary of Counts

In order to visualise the words associated to each component, the "wordcloud" CRAN package in R was used, with the "wordcloud" function [30]. The "clouds" are shown in figure 9 whereby the size of the word corresponds to the frequency. Interestingly, the clustering performs surprisingly well and those familiar with the news on 14/04/2016 will be able to pick out stories in each of the 11 clusters.

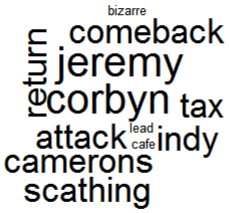
Cluster	Topic
1	South China Sea Dispute
2	Cameron/ Corbyn Tax
3	Jeremy Corbyn and EU Referendum
4	Refugee Crisis (in German)
5	Siberian Putin Cafe
6	Putin and the Panama Papers (in German)
7	Sharing Hardship in Russia
8	Child Sex Abuse
9	Rolf Harris
10	Trump and Megyn Kelly
11	Trump's running-mate

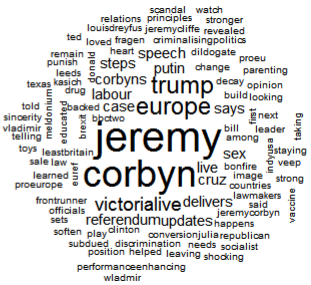
Table 7: Subjective Summary of Component Topics

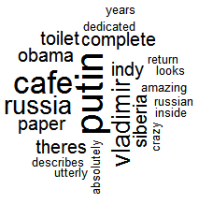
A subjective summary of what stories each cluster is associated to is shown above, however, further colour is provided here. Cluster 3 appears to be an assortment of many stories which the topic model has not separated very well. This may be due to similar word usage, that the algorithm has not been run for long enough or indicate that the hyperparameters may need fine tuning.

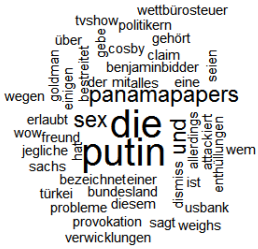
Some tweets were sourced from German news-outlets. As the words "putin" and "migrant" are the same in English as German, the clustering also picked up German clusters. In particular cluster 4 appears to relate to the global refugee crisis as it includes words such as "migrant", "asylsuchender" (asylum seeker) and "fluchtling" (refugee). Cluster 6 appears to be related to President Putin and the "panamapapers" story. Surprisingly, clusters 10 and 11 have managed to distinguish two stories on Presidential candidate Donald Trump.

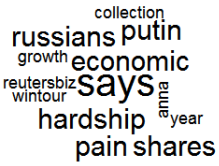
Formal cluster measures have not been carried out, as the aim of this section is simply to show how the Hierarchical Dirichlet Process Topic Model may be used on real data, for practical purposes. It is quite clear that the clustering has indeed "worked" to some degree, and particular stories have been identified. It is now possible to choose a topic and order the tweets based on the percentage of words in the tweet relating to the topic.

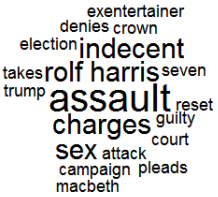


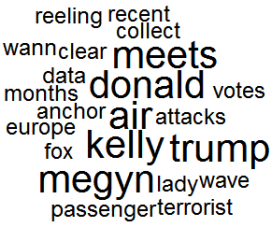












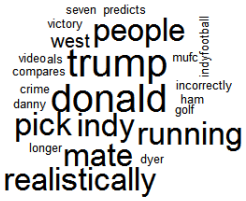


Figure 9: Word Clouds

9 Conclusion & Future Work

This dissertation has presented an accessible explanation of the Markov Chain Monte Carlo (MCMC) algorithms presented by Radford Neal in [3]. The performance of these MCMC procedures has been investigated briefly and the MCMC procedures were then developed for use in extensions of the Dirichlet Process Mixture Model. These extensions show the flexibility of the Dirichlet Process in the sense it may be used for both classification and clustering of both continuous and discrete data. Examples of these applications have been illustrated in this dissertation using interesting data-sets. Classification was performed on continuous wine data and clustering was performed on discrete data involving topic clustering amongst "tweets".

It can be seen that the Dirichlet Process (DP) is an interesting mathematical object, and that MCMC procedures allow the DP to be used in a variety of applications. The noticable areas for improvement are to investigate more efficient MCMC algorithms as the algorithms investigated here were quite slow. In addition, although some extensions of the DP mixture model were investigated here, this can be taken further by adding additional prior distributions into the DP mixture model framework, such as a prior on concentration parameters. This is likely to significantly improve the classification performance of the DP multinomial logistic classifier for example, as shown in [13]. Alternatively, other random distributions may be used instead of the Dirichlet Process. Another area that directly follows from this dissertation would be to apply the more complex algorithms, in particular algorithm 8, to the Hierarchical Dirichlet Process Mixture Model.

10 Appendices

10.1 Stationarity Proofs

Algorithm 2 Stationarity

The following supplements the explanation given in section 5.

The indicator functions for $\{\phi_k^*\}_k$ have been removed in the interest of clarity. Using what was shown above:

$$P(\phi_{1:m}^*, z_{1:n}|y_{1:n}) = \frac{\left(\prod_{j=1}^n f(y_j|\phi_{z_j}^*)\right)\left(\prod_{k=1}^{m^{(i-1)}} g_0(\phi_k^*)\right)(P(z_1,\ldots,z_n))}{\int\ldots\int\left(\prod_{j=1}^n f(y_j|\phi_{z_j}^*)\right)\left(\prod_{k=1}^{m^{(i-1)}} g_0(\phi_k^*)\right)(P(z_1,\ldots,z_n))\,d\phi_{1:m}^* dz_{1:m}}$$

Let $c_1^{-1} = \int\ldots\int\left(\prod_{j=1}^n f(y_j|\phi_{z_j}^*)\right)\left(\prod_{k=1}^{m^{(i-1)}} g_0(\phi_k^*)\right)(P(z_1,\ldots,z_n))\,d\phi_{1:m}^* dz_{1:m}$
Excahangeability of the z_i is inherited from the corresponding θ_i in the alternate representation. This allows one to write:

$$P(z_1,\ldots,z_n) = P(z_i|z_{-i})P(z_{-i}) = P(z_{-i})\left(\frac{\sum_{k=1}^{m_{-i}} n_{-i,k}\mathbb{I}_{\{k\}}(z_i)}{\alpha+n-1} + \frac{\alpha\mathbb{I}_{\{m_{-i}+1\}}(z_i)}{\alpha+n-1}\right)$$

Note, constants of proportionality that are not dependent on z_i or z_i' can be ignored. If $n_{-i,z_i}=0$, then y_i is the only observable associated to the component parameter , hence $g_0(\phi_{z_i})$ cannot be grouped into a constant of proportionality. Therefore, if $z_i = m_{-i}+1$, then z_i is a singleton and a new parameter is drawn, hence the $g_0(\phi_{z_i}^*)$ in the equation below rather than in the constant of proportionality.

$$\left(\prod_{k=1}^{m^{(i-1)}} g_0(\phi_k^*)\right)(P(z_i|z_{-i})P(z_{-i})) = P(z_{-i})\left(\prod_{k=1}^{m_{-i}} g_0(\phi_k^*)\right)\left(\frac{\sum_{k=1}^{m_{-i}} n_{-i,k}\mathbb{I}_{\{k\}}(z_i)}{\alpha+n-1} + \frac{\alpha g_0(\phi_{z_i}^*)\mathbb{I}_{\{m_{-i}+1\}}(z_i)}{\alpha+n-1}\right)$$

Let $c_2 = P(z_{-i})\left(\prod_{k=1}^{m_{-i}} g_0(\phi_k^*)\right)$and $c_3 = \prod_{j:j\neq i} f(y_j|\phi_{z_j}^*)$ then:

$$\begin{aligned} P(\phi_{1:m}^*, z_{1:n}|y_{1:n}) &= c_1c_2c_3 f(y_i|\phi_{z_i}^*)\left(\frac{\sum_{k=1}^{m_{-i}} n_{-i,k}\mathbb{I}_{\{k\}}(z_i)}{\alpha+n-1} + \frac{\alpha g_0(\phi_{z_i}^*)\mathbb{I}_{\{m_{-i}+1\}}(z_i)}{\alpha+n-1}\right) \\ &= \frac{c_1c_2c_3}{\alpha+n-1}\left(\sum_{k=1}^{m_{-i}} n_{-i,k} f(y_i|\phi_k^*)\mathbb{I}_{\{k\}}(z_i) + \alpha f(y_i|\phi_{z_i}^*)g_0(\phi_{z_i}^*)\mathbb{I}_{\{m_{-i}+1\}}(z_i)\right) \end{aligned}$$

Similarly, suppressing the indicator functions except those for z_i gives:

$$K_Z^{(i)}\left((z_i,\phi_{1:m}^*),(z_i',\phi_{1:m'}^*)\right) = b^{-1}\left(\sum_{k=1}^{m_{-i}} n_{-i,k} f(y_i|\phi_k^*)\mathbb{I}_{\{k\}}(z_i') + \alpha g_0(\phi_{new}^*) f(y_i|(\phi_{new}^*)\mathbb{I}_{\{m_{-i}+1\}}(z_i')\right)$$

Hence,

$$\begin{aligned} bK_Z^{(i)}\left((z_i',\phi_{1:m'}^*),(z_i,\phi_{1:m}^*)\right) &= \frac{\alpha+n-1}{c_1c_2c_3}P(\phi_{1:m}^*, z_{1:n}|y_{1:n}) \\ bK_Z^{(i)}\left((z_i,\phi_{1:m}^*),(z_i',\phi_{1:m'}^*)\right) &= \frac{\alpha+n-1}{c_1c_2c_3}P(\phi_{1:m'}^*, z_{-i}, z_i'|y_{1:n}) \\ P(\phi_{1:m}^*, z_{1:n}|y_{1:n})K_Z^{(i)}\left((z_i,\phi_{1:m}^*),(z_i',\phi_{1:m'}^*)\right) &= P(\phi_{1:m'}^*, z_{-i}, z_i'|y_{1:n})K_Z^{(i)}\left(\left(z_i',\phi_{1:m'}^*\right),(z_i,\phi_{1:m}^*)\right) \end{aligned}$$

This shows that the update $K_Z^{(i)}\left((z_i,\phi_{1:m}^*),(z_i',\phi_{1:m'}^*)\right)$ satisfies detailed balance and is reversible. Therefore, the chain generated from $K_Z^{(i)}$ has the correct stationary distribution, hence $K_Z\left((z_{1:n},\phi_{1:m}^*),(z_{1:n}',\phi_{1:m'}^*)\right)$ has the correct stationary distribution by the property of cycles.

Algorithm 8: Augmented Kernel Stationarity

The following supplements the explanation given in section 5.
To show that the kernel exhibits detailed balance, it is sufficient also to show proportionality, where the constant of proportionality is not dependent on the transitioning variables.
Let $T_1\left((z_i,m),(z_i',m')\right) = P(z_i,z_{-i},\phi^*_{1:m},\varphi^{(i)}_{M(z_i)}|y_{1:n})K^{(i)}_{Aug}\left(\left(z_i,\phi^*_{1:m},\varphi^{(i)}_{1:M(z_i)}\right),\left(z_i',\phi^*_{1:m'},\varphi^{(i)}_{1:M(z_i)}\right)\right)$
Case 1: z_i singleton, z_i' non-singleton
$m^{'} = m_{-i} = m-1$, $\varphi^{(i)}_1 = \phi^*_{z_i}$,$z_i' = k$

$$\begin{aligned} T_1\left((z_i,m),(z_i',m')\right) &\propto \left(f(y_i|\phi^*_{z_i})\frac{\alpha}{M}\prod_{r=1}^{M}g_0(\varphi^{(i)}_r)\right)\left(n_{-i,k}f(y_i|\phi^*_{z_i'})\mathbb{I}_{\{m_{-i}\}}(m^{'})\right) \\ &= \left(n_{-i,k}f(y_i|\phi^*_{z_i'})\prod_{r=1}^{M}g_0(\varphi^{(i)}_r)\right)\left(\frac{\alpha}{M}f(y_i|\varphi^{(i)}_1)\mathbb{I}_{\{m_{-i}+1\}}(m)\right) \\ &\propto T_1\left((z_i',m'),(z_i,m)\right) \end{aligned}$$

Case 2: z_i singleton, z_i' singleton
$\varphi^{(i)}_1 = \phi^*_{z_i}$,$m^{'} = m_{-i}+1 = m$, $\phi^*_{z_i'} = \varphi^{(i)}_j$ for some j

$$\begin{aligned} T_1\left((z_i,m),(z_i',m')\right) &\propto \left(f(y_i|\phi^*_{z_i})\frac{\alpha}{M}\prod_{r=1}^{M}g_0(\varphi^{(i)}_r)\right)\left(\frac{\alpha}{M}f(y_i|\varphi^{(i)}_j)\mathbb{I}_{\{m_{-i+1}\}}(m^{'})\right) \\ &= \left(f(y_i|\varphi^{(i)}_1)\frac{\alpha}{M}\prod_{r=1}^{M}g_0(\varphi^{(i)}_r)\right)\left(\frac{\alpha}{M}f(y_i|\phi^*_{z_i'})\mathbb{I}_{\{m_{-i}+1\}}(m^{'})\right) \\ &\propto \left(\frac{\alpha}{M}f(y_i|\phi^*_{z_i'})\prod_{r=1}^{M}g_0(\varphi^{(i)}_r)\right)\left(\frac{\alpha}{M}f(y_i|\varphi^{(i)}_1)\mathbb{I}_{\{m_{-i}+1\}}(m)\right) \\ &\propto T_1\left((z_i',m'),(z_i,m)\right) \end{aligned}$$

Case 3: z_i non-singleton, z_i' non-singleton
$m^{'} = m_{-i} = m$, $z_i = k$, $z_i' = k'$

$$\begin{aligned} T_1\left((z_i,m),(z_i',m')\right) &\propto \left(f(y_i|\phi^*_k)n_{-i,k}\prod_{r=1}^{M}g_0(\varphi^{(i)}_r)\right)\left(n_{-i,k'}f(y_i|\phi^*_{k'})\mathbb{I}_{\{m_{-i}\}}(m^{'})\right) \\ &= \left(f(y_i|\phi^*_{k'})n_{-i,k'}\prod_{r=1}^{M}g_0(\varphi^{(i)}_r)\right)\left(n_{-i,k}f(y_i|\phi^*_k)\mathbb{I}_{\{m_{-i}\}}(m)\right) \\ &\propto T_1\left((z_i',m'),(z_i,m)\right) \end{aligned}$$

Case 4: z_i non-singleton, z_i' singleton
$m^{'} = m_{-i}+1 = m+1$, $z_i = k$, $\phi^*_{z_i'} = \varphi^{(i)}_j$ for some j

$$\begin{aligned} T_1\left((z_i,m),(z_i',m')\right) &\propto \left(f(y_i|\phi^*_k)n_{-i,k}\prod_{r=1}^{M}g_0(\varphi^{(i)}_r)\right)\left(\frac{\alpha}{M}f(y_i|\varphi^{(i)}_j)\mathbb{I}_{\{m_{-i}+1\}}(m^{'})\right) \\ &\propto \left(f(y_i|\phi^*_{z_i'})\frac{\alpha}{M}\prod_{r=1}^{M}g_0(\varphi^{(i)}_r)\right)\left(n_{-i,k}f(y_i|\phi^*_k)\mathbb{I}_{\{m_{-i}\}}(m)\right) \\ &\propto T_1\left((z_i',m'),(z_i,m)\right) \end{aligned}$$

After re-introducing the appropriate normalising constant, detailed-balance is therefore satisfied.

$$P(z_i, z_{-i}, \phi^*_{1:m}, \varphi^{(i)}_{M(z_i)}|y_{1:n})K^{(i)}_{Aug}\left(\left(z_i, \phi^*_{1:m}, \varphi^{(i)}_{1:M(z_i)}\right), \left(z'_i, \phi^*_{1:m'}, \varphi^{(i)}_{1:M(z_i)}\right)\right)$$

$$= P(z'_i, z_{-i}, \phi^*_{1:m'}, \varphi^{(i)}_{M(z_i)}|y_{1:n})K^{(i)}_{Aug}\left(\left(z'_i, \phi^*_{1:m'}, \varphi^{(i)}_{1:M(z_i)}\right), \left(z_i, \phi^*_{1:m}, \varphi^{(i)}_{1:M(z_i)}\right)\right)$$

10.2 Topic Model Tweets

Number	Tweet Text (No word order)
1	american began china conducting forces joint last month naval patrols philippine sea south
2	china cites critical gun human report retort rights us violence
3	china crime decided golf longer
4	china officials punish scandal vaccine
5	begin china joint patrols philippines sea south us
6	corbyn europe happens jeremy loved play politics principles so
7	attack camerons comeback corbyn indy jeremy return rt scathing tax
8	case corbyn eu jeremy said socialist staying strong
9	attack camerons comeback corbyn indy jeremy return rt scathing tax
10	bonfire corbyn eu jeremy lead leaving rights
11	attack camerons comeback corbyn indy jeremy return rt scathing tax
12	attack camerons comeback corbyn indy jeremy return rt scathing tax
13	advice brexit corbyn given helped hes jeremy jeremycliffe labour oppose persuade recently rt told varoufakis yanis
14	bbctwo corbyn jeremy victorialive watch
15	britain corbyn eu euref jeremy labour leader position says sets stronger
16	corbyn delivers eu jeremy live proeurope referendum speech updates
17	corbyn delivers eu jeremy live proeu referendum speech updates
18	change critical eu europe i jeremycorbyn needs remain rt shortcomings victorialive victorialive
19	conversion corbyns eu jeremy sincerity telling the
20	corbyns europe opinion passion subdued
21	air attacks collect data europe months passenger recent reeling terrorist votes wave
22	countries educated europe least the
23	anna chic collection describes editor fashion kanye migrant vogue west wintour
24	asylsuchender begriffsklärung flüchtling migrant oder
25	asylsuchender begriffserklärung des doch eine flüchtling flüchtlingsstroms jkwerner migrant oder rt welt welthh zeiten
26	asylsuchender begriffserklärung eine flüchtling migrant oder rt via welt welthh
27	asylsuchender begriffsklärung flüchtling migrant oder
28	attackiert benjaminbidder der die gehört goldman panamapapers putin rt sachs sz sz usbank wegen wem wow
29	economic hardship pain putin reutersbiz rt russians says shares
30	bestreitet die die eine enthüllungen jegliche panamapapers provokation putin sagt seien über und verwicklungen
31	drug meldonium performanceenhancing putin says vladimir
32	cafe crazy dedicated indy inside putin rt russian vladimir
33	bizarre cafe indy putin rt russia theres utterly vladimir
34	first fragen ihre lady putin sie stellen vor wann wladmir
35	cafe complete obama paper putin russia theres toilet vladimir
36	allerdings als bezeichnet die einer einigen es freund gebe hat in mit politikern probleme putin türkei tvshow
37	economic hardship pain putin russians says shares
38	growth next putin return russia says year
39	absolutely amazing cafe looks putin siberia there
40	cafe complete indy obama paper putin rt siberia there toilet
41	abuse bid bill child claim cosby dismiss judge sex weighs
42	goblins macbeth national scotland sex squaddies theatre years
43	alles bundesland diesem erlaubt in ist sex und wettbürosteuer
44	backed criminalising cruz dildogate indyusa law rt sale sex ted texas toys
45	assault attack charges denies harris indecent rolf sex
46	a allowed baby child conviction despite father live offences sex
47	abuse backlash comments fry responded sex stephen survivor
48	case decay discrimination heart leeds revealed sex shocking the
49	australia baby child conviction despite fathers gammys offences sex sister stay
50	assault assault charges court crown exentertainer guilty harris indecent pleads rolf seven sex
51	donald frontrunner image republican soften steps taking trump
52	donald indy mate people pick realistically rt running trump
53	air anchor clear donald fox kelly meets megyn trump
54	clinton electionyear julia louisdreyfus trump veep
55	cruz kasich learned parenting trump what
56	campaign election reset steps takes trump
57	donald indy mate people pick realistically rt running seven trump
58	among build lawmakers looking relations trump us
59	donald mate people pick realistically running trump
60	compares danny donald dyer ham incorrectly indyfootball mufc predicts rt trump victory video west
61	donald kelly meets megyn trump
62	donald indy mate people pick realistically rt running trump

10.3 R-Code

The R-code implementations of the Neal algorithms, DPMNL and Marginalised Posterior CRF used in the HDP Topic Model is supplementary material available online at:

- https://www.dropbox.com/sh/6fm0u9mr3sq68ml/AAAVQHK9xg2czyN5FuFAD7wqa?dl=0

11 References

[1] Ferguson TS. A Bayesian Analysis of Some Nonparametric Problems. The Annals of Statistics. 1973;1(2):209–230.

[2] Sethuraman J. A Constructive Definition of Dirichlet Priors. Statistica Sinica. 1994;4(2):639–650.

[3] Neal RM. Markov Chain Sampling Methods for Dirichlet Process Mixture Models. Journal of Computational and Graphical Statistics. 2000;9(2):249–265.

[4] Papaspiliopoulos O, Roberts GO. Retrospective Markov Chain Monte Carlo Methods for Dirichlet Process Hierarchical Models. Biometrika. 2008;95(1):169–186.

[5] Teh YW. Dirichlet Process. Encyclopedia of Machine Learning. 2010;p. 280– 287.

[6] Teh YW, Jordan MI, Beal MJ, Blei DM. Hierarchical Dirichlet Processes. Journal of the American Statistical Association. 2006;101(476):1566–1581.

[7] Blackwell D, MacQueen JB. Ferguson Distributions Via Polya Urn Schemes. The Annals of Statistics. 1973;1(2):353–355.

[8] Ewens WJ. Population genetics theory-the past and the future. In: Mathematical and statistical developments of evolutionary theory. Springer; 1990. p. 177–227.

[9] Aldous D. Exchangeability and related topics. vol. 1117 of Lecture Notes in Mathematics. Springer Berlin Heidelberg; 1985. p. 1–198.

[10] Gelfand AE, Kottas A. A Computational Approach for Full Nonparametric Bayesian Inference under Dirichlet Process Mixture Models. Journal of Computational and Graphical Statistics. 2002;11(2):289–305.

[11] Roberts GO, Rosenthal JS. General state space Markov chains and MCMC algorithms. Probab Surveys. 2004;1:20–71.

[12] Hannah LA, Blei DM, Powell WB. Dirichlet Process Mixtures of Generalized Linear Models. The Journal of Machine Learning Research. 2011;12:1923–1953.

[13] Shahbaba B, Neal R. Nonlinear Models using Dirichlet Process Mixtures. The Journal of Machine Learning Research. 2009;10:1829–1850.

[14] Lichman M. UCI Machine Learning Repository; 2013.

[15] Zhao Y. R and Data Mining Workshop for the Master of Business Analytics Course; 2015.

[16] Robert C, Casella G. Monte Carlo Statistical Methods. Springer Science & Business Media; 2013.

[17] Hastings WK. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. Biometrika. 1970;57(1):97–109.

[18] Orbanz P, Teh YW. Bayesian Nonparametric Models. In: Encyclopedia of Machine Learning. Springer; 2010. .

[19] Teh YW, Jordan MI. Hierarchical Bayesian Nonparametric Models with Applications. In: Hjort N, Holmes C, Müller P, Walker S, editors. Bayesian Nonparametrics: Principles and Practice. Cambridge University Press; 2010. .

[20] Alvarez SA. An exact analytical relation among recall, precision, and classification accuracy in information retrieval. Boston College, Boston, Technical Report BCCS-02-01. 2002;p. 1–22.

[21] Gentry J. twitteR: R Based Twitter Client; 2015. R package version 1.1.9. http://CRAN.R-project.org/package=twitteR.

[22] Feinerer I, Hornik K. tm: Text Mining Package; 2015. R package version 0.6-2. http://CRAN.R-project.org/package=tm.

[23] Feinerer I, Hornik K, Meyer D. Text Mining Infrastructure in R. Journal of Statistical Software. 2008 March;25(5):1–54.

[24] Urbanek S. base64enc: Tools for base64 encoding; 2015. R package version 0.1-3. http://CRAN.R-project.org/package=base64enc.

[25] Wickham H. httr: Tools for Working with URLs and HTTP; 2015. R package version 1.0.0. http://CRAN.R-project.org/package=httr.

[26] Gentry J, Lang DT. ROAuth: R Interface For OAuth; 2015. R package version 0.9.6. http://CRAN.R-project.org/package=ROAuth.

[27] Plummer M, Best N, Cowles K, Vines K. CODA: Convergence Diagnosis and Output Analysis for MCMC. R News. 2006;6(1):7–11.

[28] Genz A, Bretz F, Miwa T, Mi X, Leisch F, Scheipl F, et al.. mvtnorm: Multivariate Normal and t Distributions; 2016. R package version 1.0-5. http://CRAN.R-project.org/package=mvtnorm.

[29] Genz A, Bretz F. Computation of Multivariate Normal and t Probabilities. Lecture Notes in Statistics. Heidelberg: Springer-Verlag; 2009.

[30] Fellows I. wordcloud: Word Clouds; 2014. R package version 2.5. http://CRAN.R-project.org/package=wordcloud.

[31] Meyer D, Dimitriadou E, Hornik K, Weingessel A, Leisch F. e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien; 2015. R package version 1.6-7. http://CRAN.R-project.org/package=e1071.

[32] Liaw A, Wiener M. Classification and Regression by randomForest. R News. 2002;2(3):18–22.

[33] Tierney L. Markov chains for exploring posterior distributions. the Annals of Statistics. 1994;p. 1701–1728.

[34] MacEachern SN, Müller P. Estimating Mixture of Dirichlet Process Models. Journal of Computational and Graphical Statistics. 1998;7(2):223–238.

[35] Escobar MD, West M. Bayesian Density Estimation and Inference Using Mixtures. Journal of the American Statistical Association. 1995;90(430):577–588.

[36] Antoniak CE. Mixtures of Dirichlet Processes with Applications to Bayesian Nonparametric Problems. The Annals of Statistics. 1974;2(6):1152–1174.

[37] Ripley BD. Stochastic Simulation. Wiley Series in Probability and Statistics. J. Wiley; 1987.

[38] ; 2016. Accessed on 14/04/2016. Author data withheld in accordance with Twitter API policy. https://twitter.com/.