Bayesian Model Based Clustering Procedures

John W. Lau[*] and Peter J. Green[†]

Department of Mathematics, University of Bristol, Bristol, UK

January 7, 2007

Abstract

This paper establishes a general formulation for Bayesian model-based clustering, in which subset labels are exchangeable, and items are also exchangeable, possibly up to covariate effects. The notational framework is rich enough to encompass a variety of existing procedures, including some recently discussed methodologies involving stochastic search or hierarchical clustering, but more importantly allows the formulation of clustering procedures that are optimal with respect to a specified loss function. Our focus is on loss functions based on pairwise coincidences, that is, whether pairs of items are clustered into the same subset or not.

Optimisation of the posterior expected loss function can be formulated as a binary integer programming problem, which can be readily solved by standard software when clustering a modest number of items, but quickly becomes impractical as problem scale increases. To combat this, a new heuristic item-swapping algorithm is introduced. This performs well in our numerical experiments, on both simulated and real data examples. The paper includes a comparison of the statistical performance of the (approximate) optimal clustering with earlier methods that are model-based but ad hoc in their detailed definition.
Some key words: Dirichlet process, Hierarchical clustering, Loss functions, Stochastic search.

1 Introduction

Clustering methods aim to separate a heterogeneous collection of items into homogeneous subsets, and are an important tool in scientific investigations in many different domains. There is a particular focus of activity in recent years, as biologists have become interested in clustering high-dimensional data generated by modern high-throughput assay techniques: for example, in order to cluster genes on the basis of gene expression measurements from microarrays. With the rapid growth in availability of computer power, Bayesian statisticians have become able to implement principled computer-intensive inferential methods for clustering, based for example on mixture models [see Fraley and Raftery (2002), Medvedovic and Sivaganesan (2002), Medvedovic et al. (2004), Yeung et al. (2001), etc.].

In a Bayesian formulation of a clustering procedure, the partition of items into subsets becomes a parameter of a probability model for the data, subject to prior assumptions, and inference about the clustering derives from properties of the posterior distribution. There are the usual opportunities for simultaneous inference about the partition and other parameters, characterising the cluster locations and spreads, for example, but our principal focus in this paper is the partition itself. In contrast to most other unknowns in a Bayesian model, where the posterior mean provides an often adequate default choice of point estimator, there is no simple natural choice for

[*]School of Statistics and Actuarial Science, University of the Witwatersrand, Private Bag 3, 2050 WITS, Johannesburg, South Africa; Email: john.w.lau@googlemail.com

[†]Department of Mathematics, University of Bristol, Bristol, BS8 1TW, United Kingdom; Email: P.J.Green@bristol.ac.uk; Tel.: (+44) (0) 117 928 7967

an estimator of a partition. Unlike the case of classical approaches, such as hierarchical clustering and K-means clustering procedures, Bayesian alternatives seem not to be fully developed.

Medvedovic and Sivaganesan (2002) and Medvedovic et al. (2004) generate a sample of partitions based on the Dirichlet process kernel mixture model, use this to estimate a pairwise similarity matrix and finally employ classical procedures [see Gordon (1999)] to define a 'best' partition. Somewhat similarly, Dubey, Hwang, et al (2004) use a Dirichlet process mixture model on 10-dimensional data derived from an analysis of protein sequences; posterior probabilities of clustering are then input to a hierarchical clustering algorithm. Again starting from a Dirichlet process kernel mixture model, but with some clever choices to expedite the computations, Dahl (2006) selects a partition from a MCMC algorithm that minimizes the sum of squared deviations from the pairwise probability matrix (of probabilities that a pair lie in the same cluster). The procedure searches for a partition subject to minimizing this objective function. Recently, Ray and Mallick (2006) developed a semi-parametric wavelet model with parameters distributed according to a Dirichlet process. They took a marginal likelihood approach, proposed by Basu and Chib (2003), to choose a partition of the data. Other relevant work, by Heard et al (2006) and by Heller and Gharamani (2005) will be discussed in Section 3.

Choosing a partition can be considered as a model choice problem, for which the usual Bayesian tools are marginal likelihoods and Bayes factors. Both of these essentially involve considering posterior probabilities under a uniform prior. This might be questioned on three grounds: the arbitrariness of taking a uniform prior, the fact that peaks of posterior probability in a high-dimensional problem can be isolated and unrepresentative, and the problem that posterior probabilities of individual partitions are difficult to compute reliably from Monte Carlo samples.

A loss function approach avoids these criticisms, and delivers an objective procedure. Binder (1978, 1981) has discussed loss functions for Bayesian clustering, although many of his choices are not appropriate for our setting, as we wish to respect exchangeability *a priori* in the labelling of clusters and of items. We therefore concentrate on loss functions defined on pairwise *coincidences*. We penalise pairs of items that are assigned to different clusters when they should be in the same one, and vice versa; the total loss is obtained by summing over all pairs. The resulting posterior expected loss is a simple linear function of posterior marginal coincidence probabilities (that is, for each pair the posterior probability they are clustered together), which are readily estimated from a Markov chain Monte Carlo (MCMC) simulation from the posterior. See also Rue (1995) for an important discussion of loss functions for high-dimensional Bayesian models with inference computed by MCMC, although his detailed proposals do not seem relevant to our context.

Section 2 introduces the Bayesian partition model under a general random measure prior. The prior and likelihood parts of the partition model are discussed separately. Section 3 discusses two existing approaches to computing a point estimate of a partition. A naive stochastic search approach and a Hierarchical Bayesian procedure are discussed and numerical comparisons are made.

We show in Section 4 that optimisation of this expected loss is a binary integer programming problem. Standard techniques are far too slow for practical use for large numbers n of items (say of the order of 1000), since the size of the integer program grows rapidly (the numbers of constraints as n^3, and of variables as n^2), and so approximations are required.

Our methods are illustrated on both simulated and real data sets. The simulated data are generated from bivariate normal mixtures with from one to four components. The real data used are the Galaxy data from Roeder (1990), and gene expression data from a leukaemia study reported in Golub et al. (1999).

2 Bayesian Clustering Models

A partition $\mathbf{p}$ separates n items with observed responses $\mathbf{y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ into subsets. Such a partition is represented as $\left\{C_1, \ldots, C_{n(\mathbf{p})}\right\}$ where C_j denotes the jth subset or cluster, for $j = 1, \ldots, n(\mathbf{p})$. Each C_j

contains indices of the items in that cluster, and the positive integer $n(\mathbf{p})$ denotes the number of clusters, which depends of course on the partition. There is a positive number e_j of items in each cluster j. Given the partition $\mathbf{p}$ of items, a model for the data is defined as

$$\pi(\mathbf{y}\,|\mathbf{p}) = \prod_{j=1}^{n(\mathbf{p})} m(\{\mathbf{y}_i, i \in C_j\}) = \prod_{j=1}^{n(\mathbf{p})} m(\mathbf{y}_{C_j}), \tag{1}$$

say, where $m(\mathbf{y}_{C_j})$ is the joint distribution of the responses for items in cluster C_j. We preserve exchangeability across cluster labels and item indexing by requiring this function not to depend on j, and, typically, to be an exchangeable function of its arguments $\{\mathbf{y}_i, i \in C_j\}$. Both these kinds of exchangeability follow usual practice in clustering problems. Naturally, the demands of specific contexts may demand variations in these assumptions; for example, in separate work we are adapting our models and criteria to gene expression profile clustering, allowing a 'background' cluster not exchangeable with the others.

Since we also wish to perform clustering in the presence of covariate information $\{\mathbf{x}_i, i = 1, 2, \ldots, n\}$ on the items, we allow $m(\mathbf{y}_{C_j})$ to depend tacitly on $\{\mathbf{x}_i, i \in C_j\}$, and exchangeability over items then relates to permuting $(\mathbf{y}_i, \mathbf{x}_i)$ pairs jointly. Typically, we suppose

$$m(\mathbf{y}_{C_j}) = \int_{\mathcal{U}} \prod_{i \in C_j} k(\mathbf{y}_i \mid u_j)\, G_0(du_j) \quad \text{or} \quad \int_{\mathcal{U}} \prod_{i \in C_j} k(\mathbf{y}_i \mid u_j; \mathbf{x}_i)\, G_0(du_j) \tag{2}$$

and $k(\mathbf{y}_i\,|u_j)$ is a density for $\mathbf{y}_i$s with parameters/latent variables u_j and G_0 is a distribution function of u_j on the space $\mathcal{U}$.

Equation (2), which guarantees the required exchangeability properties, may be viewed either as a formal representation [de Finetti (1930, 1974), Hewitt and Savage (1955) and Diaconis and Freedman (1984, 1987)], or as an ingredient in the specification of a Bayesian hierarchical model, in which clusters are assigned characterising parameters u_j, i.i.d. from distribution G_0, and item responses within clusters drawn independently from distribution $k(\mathbf{y}_i \mid u_j)$ or $k(\mathbf{y}_i \mid u_j; \mathbf{x}_i)$.

In the full Bayesian formulation, a prior probability would be assigned to each partition $\mathbf{p}$, leading to a posterior of the form

$$\pi(\mathbf{p} \mid \mathbf{y}) \propto \phi(\mathbf{p}) = \pi(\mathbf{p}) \prod_{j=1}^{n(\mathbf{p})} m(\mathbf{y}_{C_j}) \tag{3}$$

[References include Hartigan(1990), Barry and Hartigan (1992), and Quintana and Iglesias (2003), who regard this as a product partition model ($\mathcal{PPM}$). See also Brunner and Lo (1999) and Lo (2005)]

Of key importance to practical inference in this model setting is the question whether $m(\mathbf{y}_{C_j})$ is explicitly known or not. When it is, MCMC procedures are greatly simplified, since cluster-specific parameters u_j can be integrated out, and the posterior distribution of $\mathbf{p}$ alone (or possibly $\mathbf{p}$ together with a few hyperparameters) can be the target of an MCMC sampler. For the remainder of this paper, we will suppose that $m(\mathbf{y}_{C_j})$ is explicitly available: in the context of (2) this amounts to assuming that G_0 is conjugate for the density $k(\mathbf{y}_i \mid \cdot)$.

2.1 The prior part of the clustering model

In the absence of real prior information about the items, we will assign positive prior probability to every possible partition. In the interests of computational convenience, we might be attracted to prior models for which posterior simulation methods are fully developed, and this leads us to models based on random probability measures. The Ferguson (1973) Dirichlet process [see also Antoniak (1974)] is a popular choice, suggested by Lo (1984). Recently, Ishwaran and James (2001, 2003a, 2003b) employ stick breaking random probability measures, including the Kingman–Pitman–Yor Poisson–Dirichlet process, as a prior for the kernel mixture model. Another familiar example is the finite mixture model with κ components, and a symmetric Dirichlet distribution with parameters $(\delta, \delta, \ldots, \delta)$ on the component weights; this yields an explicit prior on the partition when the weights are integrated out. For these models, the corresponding $\pi(\mathbf{p})$s are

Random Probability Measures	Parameters	$\pi(\mathbf{p}) \propto$
Dirichlet process	$\theta > 0$	$\theta^{n(\mathbf{p})} \prod_{j=1}^{n(\mathbf{p})} \Gamma(e_j)$
Finite mixture model, Dirichlet weights	$\kappa, \delta > 0$	$\frac{\kappa!}{(\kappa - n(\mathbf{p}))!} \prod_{j=1}^{n(\mathbf{p})} \frac{\Gamma(\delta + e_j)}{\Gamma(\delta)}$
Poisson–Dirichlet process	$\theta > -\alpha, 0 \leq \alpha < 1$	$\prod_{j=1}^{n(\mathbf{p})} \left[(\theta + \alpha(j-1)) \frac{\Gamma(e_j - \alpha)}{\Gamma(1-\alpha)} \right]$

James (2002, 2005) and Lijoi et al. (2005) consider more general construction of kernel mixtures based on certain classes of random probability measures. For more random probability measures, we refer to Kingman (1975, 1993), Pitman (1995, 1996, 2003) as well as Pitman and Yor (1997).

In fact, however, in the conjugate setting, posterior simulation is equally straightforward for much wider classes of prior, and all that is needed for implementation is the availability of the ratio of prior probabilities $\pi(\mathbf{p}')/\pi(\mathbf{p})$ when $\mathbf{p}'$ is obtained from $\mathbf{p}$ by re-assigning a single item. Thus, given prior information, we are free to assign prior probabilities $\pi(\mathbf{p})$ to suit the situation; the procedures discussed in this paper will still work for most choices.

2.2 The data part of the clustering model

We discuss two kinds of model (2). One of them follows standard Bayesian modelling lines, taking the density k of (2) to be normal with unknown mean and unknown variance/precision, and G_0 as the (Normal–Gamma) parameter prior. Then $m(\mathbf{y}_{C_j})$ will be a t-density. For the other kind of model, we directly assign $m(\mathbf{y}_{C_j})$ to be an exchangeable density of $\mathbf{y}_{C_j}$. This can be a convenient way to proceed, and also allows us to make connections with the classical theory of clustering.

Normal–Gamma set up. Motivated by application to clustering gene expression profiles, we discuss a regression type model that expresses a vector response in terms of a linear combination of known covariates. We take $\mathbf{y}_i$ as a vector, written as $\mathbf{y}_i = [y_{i1} \cdots y_{iS}]'$ for $i = 1, \ldots, n$. Given the covariates $\mathbf{x}_s = [x_{s1} \cdots x_{sK}]'$ and that i lies in the cluster j, the parameters/latent variables are $\boldsymbol{\beta}_j = [\beta_{j1} \cdots \beta_{jK}]'$ and τ_j. The regression part of the model can be written as

$$\mathbf{y}_i = \begin{bmatrix} y_{i1} \\ \vdots \\ y_{iS} \end{bmatrix} = \sum_{k=1}^{K} \beta_{jk} \begin{bmatrix} x_{1k} \\ \vdots \\ x_{Sk} \end{bmatrix} + \begin{bmatrix} \epsilon_{i1} \\ \vdots \\ \epsilon_{iS} \end{bmatrix} = [\mathbf{x}_1 \cdots \mathbf{x}_S]' \boldsymbol{\beta}_j + \boldsymbol{\epsilon}_i \tag{4}$$

where $\boldsymbol{\epsilon}_i \sim N\left(\mathbf{0}_{S\times 1}, \tau_j^{-1} \mathbf{I}_{S\times S}\right)$, where $\mathbf{0}_{S\times 1}$ is the S-dimensional zero vector and $\mathbf{I}_{S\times S}$ is the $S \times S$ identity matrix. The covariates $[\mathbf{x}_1 \cdots \mathbf{x}_S]'$ are fixed and observed, and may as usual include dummy covariates for factor levels. It could be an identity matrix (when $K = S$). From (4), $\mathbf{y}_i$ follows a multivariate Normal density with mean $\boldsymbol{\beta}_j$ and variance $\tau_j^{-1} \mathbf{I}_{S\times S}$. Writing $u_j = (\boldsymbol{\beta}_j, \tau_j)$, the kernel is represented as $k\left(\mathbf{y}_i \left| \boldsymbol{\beta}_j, \tau_j\right.\right)$, a multivariate normal density, $N\left([\mathbf{x}_1 \cdots \mathbf{x}_S]' \boldsymbol{\beta}_j, \tau_j^{-1} \mathbf{I}_{S\times S}\right)$. To complete the normal set up, given τ_j, $\boldsymbol{\beta}_j$ follows the K-dimensional multivariate normal with mean $\mathbf{m}_0$ and variance $(\tau_j \mathbf{t}_0)^{-1}$, and τ_j follows the univariate Gamma with shape a_0 and scale b_0. We denote the joint distribution $G_0(d\boldsymbol{\beta}_j, d\tau_j)$ as a joint Gamma–Normal distribution, Gamma–Normal $(a_0, b_0, \mathbf{m}_0, \mathbf{t}_0)$. Based on the set up, we have

$$m\left(\mathbf{y}_{C_j}\right) = \frac{\Gamma\left(\dfrac{2a_0 + e_j S}{2}\right)}{\Gamma\left(\dfrac{2a_0}{2}\right)} \frac{\left|\dfrac{b_0}{a_0}\left(\mathbf{X}_{C_j}\mathbf{t}_0^{-1}\mathbf{X}'_{C_j} + \mathbf{I}_{e_jS\times e_jS}\right)\right|^{-1/2}}{(2a_0\pi)^{e_jS/2}} \tag{5}$$
$$\times\left(1 + \frac{1}{2a_0}\left(\mathbf{y}_{C_j} - \mathbf{X}_{C_j}\mathbf{m}_0\right)'\left(\frac{b_0}{a_0}\left(\mathbf{X}_{C_j}\mathbf{t}_0^{-1}\mathbf{X}'_{C_j} + \mathbf{I}_{e_jS\times e_jS}\right)\right)^{-1}\left(\mathbf{y}_{C_j} - \mathbf{X}_{C_j}\mathbf{m}_0\right)\right)^{-(2a_0+e_jS)/2}$$

where $\mathbf{y}_{C_j} = \left[\mathbf{y}'_{i_1}\cdots\mathbf{y}'_{i_{e_j}}\right]'$ and $\mathbf{X}_{C_j} = [[\mathbf{x}_1\cdots\mathbf{x}_S]\cdots[\mathbf{x}_1\cdots\mathbf{x}_S]]'$ for $C_j = \{i_1, \ldots, i_{e_j}\}$. Note that $\mathbf{y}_{C_j}$ is a e_jS vector and $\mathbf{X}_{C_j}$ is a $e_jS \times K$ matrix. Moreover, $m\left(\mathbf{y}_{C_j}\right)$ is a multivariate t density with mean $\mathbf{X}_{C_j}\mathbf{m}_0$, scale $\dfrac{b_0}{a_0}\left(\mathbf{X}_{C_j}\mathbf{t}_0^{-1}\mathbf{X}'_{C_j} + \mathbf{I}_{e_jS\times e_jS}\right)$ and degrees of freedom $2a_0$.

Directly assigned marginals approach. Alternatively, we can directly assign $m\left(\mathbf{y}_{C_j}\right)$ to be some exchangeable (symmetric) function, that is non-negative and integrable. One interesting construction is

$$m\left(\mathbf{y}_{C_j}\right) = e^{-\psi\left(\mathbf{y}_{C_j}\right)} \tag{6}$$

where the function ψ is non-negative and exchangeable. A popular choice of function ψ is

$$\psi\left(\mathbf{y}_{C_j}\right) = w_j \times \sum_{i\in C_j}\left(\mathbf{y}_i - \bar{\mathbf{y}}_{C_j}\right)'\left(\mathbf{y}_i - \bar{\mathbf{y}}_{C_j}\right) \tag{7}$$

where

$$\bar{\mathbf{y}}_{C_j} = \frac{1}{e_j}\sum_{i\in C_j}\mathbf{y}_i$$

Usually, we would like to choose $w_j = 1$ and the ψ is related to the sum of squared errors within each cluster. This gives a stochastic version of the Ward (1963) clustering model. Moreover, $w_j = e_j$ gives us an interesting case, that is

$$\psi\left(\mathbf{y}_{C_j}\right) = e_j \times \sum_{i\in C_j}\left(\mathbf{y}_i - \bar{\mathbf{y}}_{C_j}\right)'\left(\mathbf{y}_i - \bar{\mathbf{y}}_{C_j}\right) = \sum_{i_1<i_2, i_1, i_2\in C_j} d_{i_1,i_2} \tag{8}$$

where

$$d_{i_1,i_2} = \left(\mathbf{y}_{i_1} - \mathbf{y}_{i_2}\right)'\left(\mathbf{y}_{i_1} - \mathbf{y}_{i_2}\right)$$

These models focus on the sum of the pairwise distances of all pairs in each cluster. See also Banfield and Raftery (1993) for some existing models. For an analogue of (5), we could take

$$\psi\left(\mathbf{y}_{C_j}\right) = \sum_{i\in C_j}\left(\mathbf{y}_i - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_j}\right)'\left(\mathbf{y}_i - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_j}\right) \tag{9}$$

where

$$\hat{\boldsymbol{\beta}}_{C_j} = \left(\mathbf{X}'_{C_j}\mathbf{X}_{C_j}\right)^{-1}\mathbf{X}'_{C_j}\mathbf{y}_{C_j} \tag{10}$$

This is an extension from the random sample case (7) to the regression case. It is also minus twice the exponent of a normal density which is a limiting case of (5), in which we take $b_0 = a_0, \mathbf{t}_0 = \mathbf{0}, \mathbf{m}_0 = \mathbf{0}$ and $2a_0 \to \infty$. Note that (10) is valid for any size of cluster if and only if $S \geq K$; otherwise estimators other than least squares could be used.

3 Clustering procedures

MCMC procedures deliver an approximation to the posterior of the partition $\mathbf{p}$, but for many purposes a single point estimate $\widehat{\mathbf{p}}$ will be sought. In the following section, we consider optimal point estimates based on loss functions, but using the posterior alone, perhaps the only reasonable estimate is the posterior mode, maximising (3). Two main kinds of approach have been considered recently, and will be discussed in the section.

First, stochastic search methods simply run a MCMC sampler in equilibrium for the posterior distribution, and record the $\mathbf{p}$ with highest posterior probability encountered during the run. For example, Brunner and Lo (1999) do this for a Dirichlet process model, using the usual Gibbs sampler for the partition allocation variables (the "weighted Chinese restaurant process" procedure). Recently, Nobile and Fearnside (2005) proposed an allocation sampler which seems to be an efficient alternative sampler for such models, although their setting is the finite mixture model with Multinomial–Dirichlet prior. Other than the sampler used here, stochastic search may be performed using other MCMC samplers over the posterior distribution. Examples include the samplers proposed by West et al. (1994) and Richardson and Green (1997) for mixture models. The chief weakness of such stochastic search schemes is that, since the number of possible partitions (given by the Bell number) is huge, it is impossible to visit all possible partitions to identify $\hat{\mathbf{p}}$. Moreover, it seems intractable to produce proper summary statistics, say frequency counts, of the visited partitions as the set of visited partitions may be very irregular. Yet, we may encounter a partition close to $\hat{\mathbf{p}}$ fortuitously. A more disciplined but computationally-intensive MCMC-based approach to optimisation is simulated annealing (Kirkpatrick, et al, 1983), but we do not pursue this here.

Heard et al. (2006) and Heller and Ghahramani (2005) have both avoided MCMC sampling in approximating the posterior mode in a Bayesian clustering model. Bayesian hierarchical clustering is a deterministic hierarchical agglomerative model based procedure. Here we follow in detail Heard et al, who use finite mixture models over a Multinomial–Dirichlet prior. The idea is simple: two clusters are combined iteratively to maximise the posterior probability (3). A sequence of partitions is built up from n singleton clusters initially, terminating in a single cluster. As the number of clusters $n(\mathbf{p})$ decreases, the values of the posterior probability (3) first rises and then falls again; the best partition over this sequence can then be discovered.

3.1 Stochastic search by posterior sampling of partitions

We briefly discuss the Gibbs sampler for allocation variables, also known as the weighted Chinese restaurant process procedure. The objective is to sample $\mathbf{p}$ from the stationary distribution $\pi(\mathbf{p}\,|\mathbf{y})$ [see Escobar and West (1995, 1998), Ishwaran and James (2001, 2003a, 2003b), Lo et al. (1996), MacEachern (1994), MacEachern and Müller (1998, 2000), Neal (2000), West et al. (1994), etc.]. Here we simply introduce the procedures. Given a partition $\mathbf{p}$, one item, say k, is removed, leaving a partition of $n-1$ items which we denote as $\mathbf{p}_{-k} = \left\{C_{1,-k}, \ldots, C_{n(\mathbf{p}_{-k}),-k}\right\}$. The size of each cluster is $e_{j,-k}$ for $j = 1, \ldots, n(\mathbf{p}_{-k})$ and the number of clusters is $n(\mathbf{p}_{-k})$. The item will be assigned to a new cluster with probability proportional to

$$m\left(\mathbf{y}_{\{k\}}\right) \times \begin{cases} \theta & \text{for the Dirichlet process} \\ (\kappa - n(\mathbf{p}_{-k}))\delta & \text{for the finite mixture model} \\ \theta + \alpha n(\mathbf{p}_{-k}) & \text{for the Poisson–Dirichlet process} \\ 1 & \text{for the Uniform Prior} \end{cases}$$

and the item will be assigned to the cluster j with probability proportional to

$$\frac{m\left(\mathbf{y}_{\{k\}\cup C_{j,-k}}\right)}{m\left(\mathbf{y}_{C_{j,-k}}\right)}\times\begin{cases} e_{j,-k} & \text{for the Dirichlet process}\\ e_{j,-k}+\delta & \text{for the finite mixture model}\\ e_{j,-k}-\alpha & \text{for the Poisson–Dirichlet process}\\ 1 & \text{for the Uniform Prior}\end{cases}$$

for $j=1,\ldots,n\left(\mathbf{p}_{-k}\right)$. In general, the sampler can be described as follows. The item will be assigned to a new cluster with probability proportional to

$$m\left(\mathbf{y}_{\{k\}}\right)\times\frac{\pi\left(C_{1,-k},\ldots,C_{n(\mathbf{p}_{-k}),-k},\{k\}\right)}{\pi\left(C_{1,-k},\ldots,C_{n(\mathbf{p}_{-k}),-k}\right)}$$

and the item will be assigned to the cluster j with probability proportional to

$$\frac{m\left(\mathbf{y}_{\{k\}\cup C_{j,-k}}\right)}{m\left(\mathbf{y}_{C_{j,-k}}\right)}\times\frac{\pi\left(C_{1,-k},\ldots,C_{j,-k}\cup\{k\},\ldots,C_{n(\mathbf{p}_{-k}),-k}\right)}{\pi\left(C_{1,-k},\ldots,C_{j,-k},\ldots,C_{n(\mathbf{p}_{-k}),-k}\right)}$$

for $j=1,\ldots,n\left(\mathbf{p}_{-k}\right)$.

As an alternative to the weighted Chinese restaurant process procedure, Nobile and Fearnside (2005) propose a series of procedures to sample from a partition model generated by a finite mixture models with a Multinomial–Dirichlet prior in order to enhance the mixing of the sampler. Inspired by the reversible jump sampling idea, it is an efficient Metropolis–Hastings sampler for re-allocating items among partitions. Here we mention two related moves. The *Absorption/Merge Move* proposes combining two randomly chosen clusters into one; the *Ejection/Split Move* is the reverse move. If the two clusters concerned are C_{j_1} and C_{j_2}, the acceptance probabilities for these moves are $\min\{1,R\}$ and $\min\{1,R^{-1}\}$, where in general,

$$R=\frac{m\left(\mathbf{y}_{C_{j_1}\cup C_{j_2}}\right)}{m\left(\mathbf{y}_{C_{j_1}}\right)m\left(\mathbf{y}_{C_{j_2}}\right)}\frac{\pi\left(C_1,\ldots,C_{j_1}\cup C_{j_2},\ldots,C_{n(\mathbf{p})}\right)}{\pi\left(C_1,\ldots,C_{n(\mathbf{p})}\right)}$$

Other moves are available by systematically transferring items between partition subsets.

3.2 Bayesian hierarchical clustering procedures

We will use our model (3) to present the idea of the deterministic hierarchical agglomerative model based clustering procedure of Heard et al. (2006), which can generally be computed very cheaply compared to methods requiring Monte Carlo sampling over partitions. There are some interesting features that only appear in partition models generated by the class of random probability measures, e.g, the Dirichlet process and Poisson–Dirichlet process. We initiate the procedure with all singleton clusters, and step-by-step, combine pairs of clusters until all items are clustered together.

The iterative step is at each stage to combine the two clusters C_{j_1} and C_{j_2} that maximise

$$\frac{m\left(\mathbf{y}_{C_{j_1}\cup C_{j_2}}\right)}{m\left(\mathbf{y}_{C_{j_1}}\right)m\left(\mathbf{y}_{C_{j_2}}\right)}\times\eta\left(e_{j_1},e_{j_2}\right)$$

where

$$\eta\left(e_{j_1}, e_{j_2}\right)=\begin{cases}\dfrac{\Gamma\left(e_{j_1}+e_{j_2}\right)}{\Gamma\left(e_{j_1}\right)\Gamma\left(e_{j_2}\right)} & \text{for the Dirichlet process}\\ \dfrac{\Gamma\left(e_{j_1}+e_{j_2}+\delta\right)}{\Gamma\left(e_{j_1}+\delta\right)\Gamma\left(e_{j_2}+\delta\right)} & \text{for the finite mixture model}\\ \dfrac{\Gamma\left(e_{j_1}+e_{j_2}-\alpha\right)}{\Gamma\left(e_{j_1}-\alpha\right)\Gamma\left(e_{j_2}-\alpha\right)} & \text{for the Poisson–Dirichlet process}\\ 1 & \text{for the Uniform Prior}\end{cases}$$

This choice of clusters to combine maximises the posterior quantity (3) among all partitions that can be obtained from the present one by merging a pair of clusters. Suppose we move $\mathbf{p}=\left\{C_1,\ldots,C_{n(\mathbf{p})}\right\}$ to $\mathbf{p}^*=\left\{C_1^*,\ldots,C_{n(\mathbf{p}^*)}^*\right\}$ where e_j for $j=1,\ldots,n\left(\mathbf{p}\right)$ and e_j^* for $j=1,\ldots,n\left(\mathbf{p}^*\right)$ are the size of clusters respectively. Then, from (3), we find:

$$\phi\left(\mathbf{p}^*\right)=\phi\left(\mathbf{p}\right)\times\left(\frac{1}{\xi}\eta\left(e_{j_1},e_{j_2}\right)\frac{m\left(\mathbf{y}_{C_{j_1}\cup C_{j_2}}\right)}{m\left(\mathbf{y}_{C_{j_1}}\right)m\left(\mathbf{y}_{C_{j_2}}\right)}\right)$$

where

$$\xi=\begin{cases}\theta & \text{for the Dirichlet process}\\ \dfrac{\left(\kappa-\left(n\left(\mathbf{p}\right)-1\right)\right)\delta}{\Gamma(\delta+1)} & \text{for the finite mixture model}\\ \dfrac{\theta+\alpha\left(n\left(\mathbf{p}\right)-1\right)}{\Gamma(1-\alpha)} & \text{for the Poisson–Dirichlet process}\\ 1 & \text{for the Uniform Prior}\end{cases}$$

In practice, Heard et al. (2006) construct a sequence of partitions from the all-singletons partition to a single cluster partition, and then determine the best partition by comparisons though (3). Note that the maximizing process does not depend on the quantity ξ.

We now consider particular cases of this procedure for specific directly assigned choices of $m(\mathbf{y}_{C_j})$. If we choose $m\left(\mathbf{y}_{C_j}\right)$ in (2), to be given by (6) and (7), and tailor-make $\pi\left(\mathbf{p}\right)\propto\theta^{n(\mathbf{p})}$ for an assumed value of θ, the procedure works as follows. Iteratively, we combine two clusters C_{j_1} and C_{j_2} if

$$\frac{e_{j_1}e_{j_2}}{e_{j_1}+e_{j_2}}\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}\right)'\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}\right)\leq\min_{j_2,j_1\in\{1,\ldots,n(\mathbf{p})\},j_1\neq j_2}\left\{\frac{e_{j_1}e_{j_2}}{e_{j_1}+e_{j_2}}\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}\right)'\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}\right)\right\};\quad(11)$$

see the Appendix for the proof. We can interpret the resulting procedure as an instance of the hierarchical clustering algorithm of Ward (1963), as the parameter θ can be regarded as an analogue of the presumed number of clusters in the classical Ward procedure. For any given θ, we can compute functionals of the prior information we are assigning to partitions through simulation.

Next we take $m\left(\mathbf{y}_{C_j}\right)$ to be (8). The procedure works as follows. Iteratively, we combine two clusters C_{j_1} and C_{j_2} if

$$\sum_{i_1\in C_{j_1}}\sum_{i_2\in C_{j_2}}d_{i_1,i_2}\leq\min_{j_2,j_1\in\{1,\ldots,n(\mathbf{p})\},j_1\neq j_2}\left\{\sum_{i_1\in C_{j_1}}\sum_{i_2\in C_{j_2}}d_{i_1,i_2}\right\}.\quad(12)$$

For the regression case, we take $m\left(\mathbf{y}_{C_j}\right)$ to be (9), and the procedure then works as follows.

$$\begin{aligned}&\left(\hat{\boldsymbol{\beta}}_{C_{j_1}}-\hat{\boldsymbol{\beta}}_{C_{j_2}}\right)'\mathbf{W}_{C_{j_1},C_{j_2}}\left(\hat{\boldsymbol{\beta}}_{C_{j_1}}-\hat{\boldsymbol{\beta}}_{C_{j_2}}\right)\\&\quad\leq\min_{j_2,j_1\in\{1,\ldots,n(\mathbf{p})\},j_1\neq j_2}\left\{\left(\hat{\boldsymbol{\beta}}_{C_{j_1}}-\hat{\boldsymbol{\beta}}_{C_{j_2}}\right)'\mathbf{W}_{C_{j_1},C_{j_2}}\left(\hat{\boldsymbol{\beta}}_{C_{j_1}}-\hat{\boldsymbol{\beta}}_{C_{j_2}}\right)\right\}\end{aligned}\quad(13)$$

where

$$\mathbf{W}_{C_{j_1},C_{j_2}}=\left[\left(\mathbf{X}_{C_{j_1}}'\mathbf{X}_{C_{j_1}}\right)^{-1}+\left(\mathbf{X}_{C_{j_2}}'\mathbf{X}_{C_{j_2}}\right)^{-1}\right]^{-1}$$

3.3 Real and simulated data sets

In order to make comparisons with different procedures, we employ 4 sets of artificial data and 2 sets of real data to illustrate our methodology.

3.3.1 Simulated mixtures

We generate 4 sets of data. We first consider a 4 component mixture of bivariate Normal densities,

$$\sum_{i=1}^{4} w_i N(\mu_i, \Sigma_i) \text{ with parameters } \begin{array}{cccc} \mu_1 = (2,2)' & \mu_2 = (2,-2)' & \mu_3 = (-2,2)' & \mu_4 = (-2,-2)' \\ \Sigma_1 = \mathbf{I}_{2\times2} & \Sigma_2 = \mathbf{I}_{2\times2} & \Sigma_3 = \mathbf{I}_{2\times2} & \Sigma_4 = \mathbf{I}_{2\times2} \end{array}$$

where $\mathbf{I}_{2\times2}$ represent the identity matrix of order 2. We generate an artificial data set with the following settings,

	w_1	w_2	w_3	w_4
Model 1	1	0	0	0
Model 2	1/2	1/2	0	0
Model 3	1/3	1/3	1/3	0
Model 4	1/4	1/4	1/4	1/4

We take the covariate $[\mathbf{x}_1 \cdots \mathbf{x}_S]' = \mathbf{I}_{2\times2}$ for $S = 2$ and $K = 2$. A comparison between the stochastic search procedures and hierarchical clustering procedures is performed. We take 100 sets of data from each of the models 1–4, each with $n = 100$ data points. We fit the data to the model (5) with the Dirichlet process prior. The parameters are assigned to be $\theta = 1, a_0 = 1, b_0 = 0.01, \mathbf{m}_0 = [0 \cdots 0]', \mathbf{t}_0 = 0.01\mathbf{I}$. We employ both SS and BH procedures. For the stochastic search method, we run the Gibbs sampler for 20000 sweeps. We record the partition of greatest posterior probability among all 20000 sweeps. In Figure 1 we plot the $\log(\phi(\mathbf{p}))$ of the stochastic search (SS) versus that of Bayesian hierarchical clustering procedures (BH). This seems to suggest that the "hidden" number of components has an effect on the performance of both procedures. The stochastic search (SS) attains higher posterior probability partitions than the Bayesian hierarchical clustering procedure (BH) when the "hidden" number of components increases.

3.3.2 Galaxy data

The famous Roeder (1990) galaxy data consists of measurements of velocities in km/sec of 82 galaxies from a survey of the Corona Borealis region. The data set is available in the paper, as well as in the **VR** package of the statistical system **R**. We take the covariate $[\mathbf{x}_1 \cdots \mathbf{x}_S]' = 1$ for $S = 1$ and $K = 1$. We fit our models to these data. We plot the log posterior probability $\log(\phi(\mathbf{p}))$ against the number of clusters of the Bayesian hierarchical clustering procedures (BH) in Figure 2, and against sweep number for the first 200 sweeps of the stochastic search (SS) in Figure 3. Interestingly, both procedures attain the same highest-posterior-probability $\mathbf{p}$, suggesting that the optimal number of clusters is 3. Moreover, stochastic search (SS) first attains the best $\mathbf{p}$ at the 181^{th} sweep of the 20000 sweeps used for comparison. Of course, we may want more sweeps to get better performance. In Figure 4, in addition to the histogram we indicate the observations, and the inferred clusters, numbered in no particular order.

3.3.3 Leukaemia data

The leukaemia data of Golub et al. (1999) consists of 7129 gene expression levels of 72 patients with either acute myeloid leukaemia (AML) or acute lymphoblastic leukaemia (ALL). Golub et al. (1999) split the data into training and test sets. The training set consists of 38 patients, of which 27 have acute lymphoblastic (ALL) and 11 have acute myeloid leukaemia (AML) cases, while the test set of 34 patients has 20 with ALL and

Figure 1: Comparisons between SS and BH under Models 1–4 based on log posterior probability.

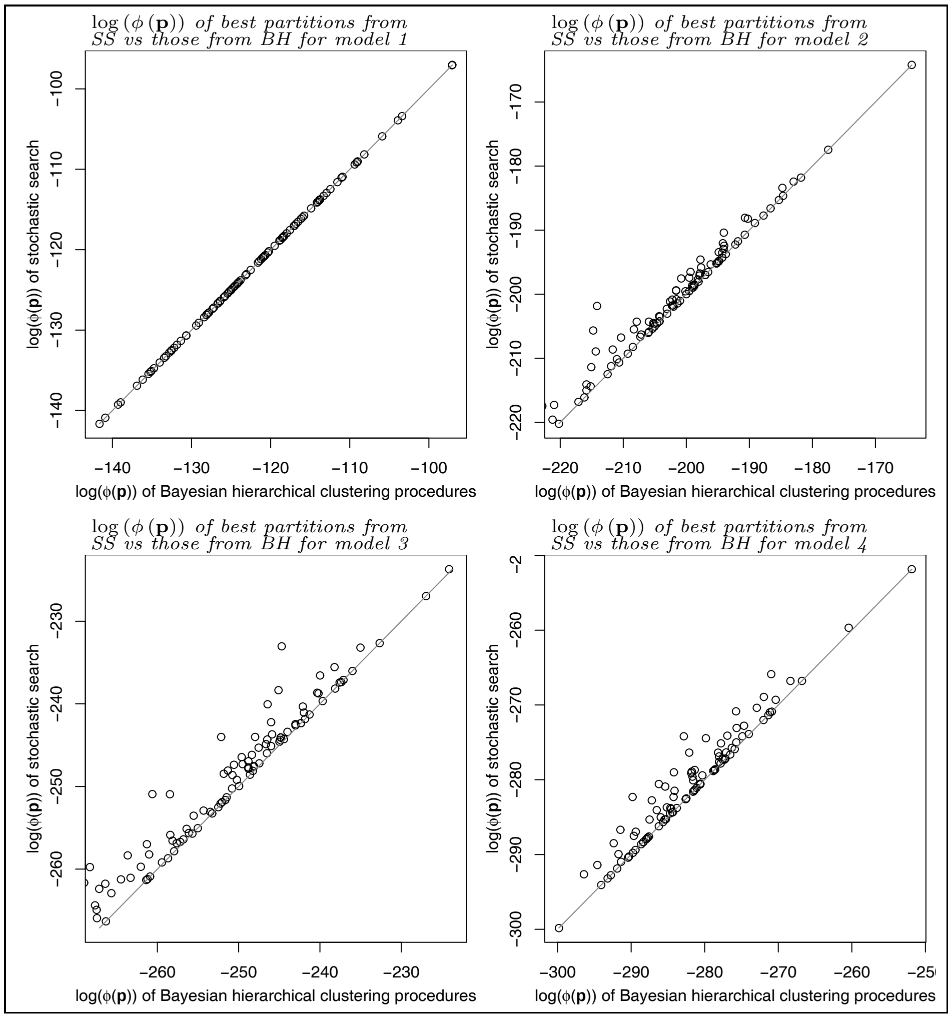


Figure 2: Log posterior partition probability vs number of clusters, using BH algorithm applied to the Galaxy data.

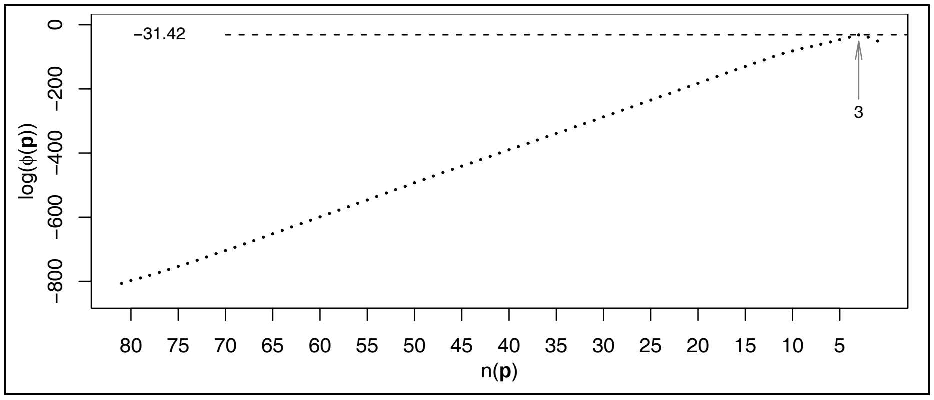


Figure 3: Log posterior partition probability vs sweep number, using SS algorithm applied to the Galaxy data.

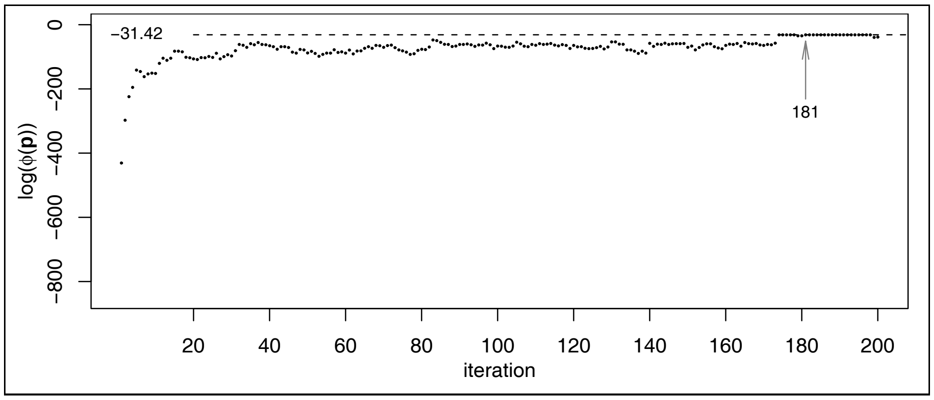


Figure 4: The best partition of the Galaxy data produced by the SS and BH algorithms.

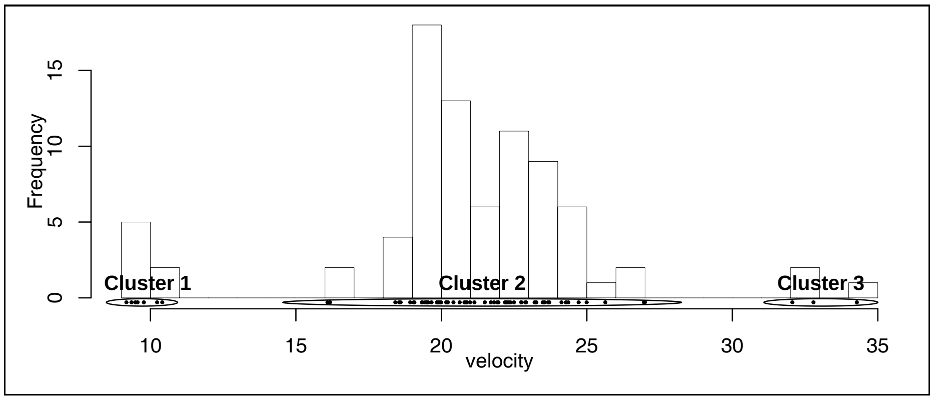
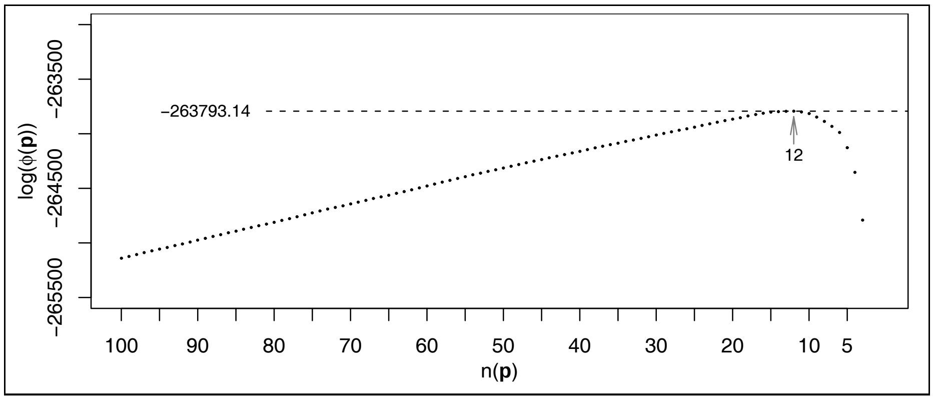


Figure 5: Log posterior partition probability vs number of clusters, using BH algorithm applied to the Leukaemia data.



14 with AML. Acute lymphoblastic leukaemia actually arises from two different types of lymphocytes (T-cell and B-cell). Among all 47 ALL patients, there are 38 and 9 with B-cell and T-cell lymphocytes respectively. The data set is available at the website `http://www-genome.wi.mit.edu/mpr/data_set_ALL_AML.html` and in the **golubEsets** package of the **Bioconductor** system (Gentleman, et al, 2004). Following Dudoit et al. (2002), to transform the data we employ: (a) thresholding using a floor of 100 and ceiling of 16,000; (b) filtering, exclusion of genes with max/min$\leq$5 or max$-$min$\leq$1600, where max and min refer to the maximum and minimum intensities for a particular gene across the 72 mRNA samples; and (c) base 10 logarithmic transformation. There are 1656 genes left. The analyzed data set is a normalized version of the selected gene expressions. Each gene is normalized by subtracting its mean and dividing by its standard deviation across patients/samples. The design matrix of covariates $[\mathbf{x}_1 \cdots \mathbf{x}_S]'$ is taken to be

$$[\mathbf{x}_1 \cdots \mathbf{x}_S]' = \underbrace{\begin{bmatrix} 1\cdots1 & 1\cdots1 & 1\cdots1 \\ 1\cdots1 & 0\cdots0 & 0\cdots0 \\ 0\cdots0 & 1\cdots1 & 0\cdots0 \end{bmatrix}'}_{\text{ALL (B-CELL)} \quad \text{ALL (T-CELL)} \quad \text{AML}}$$

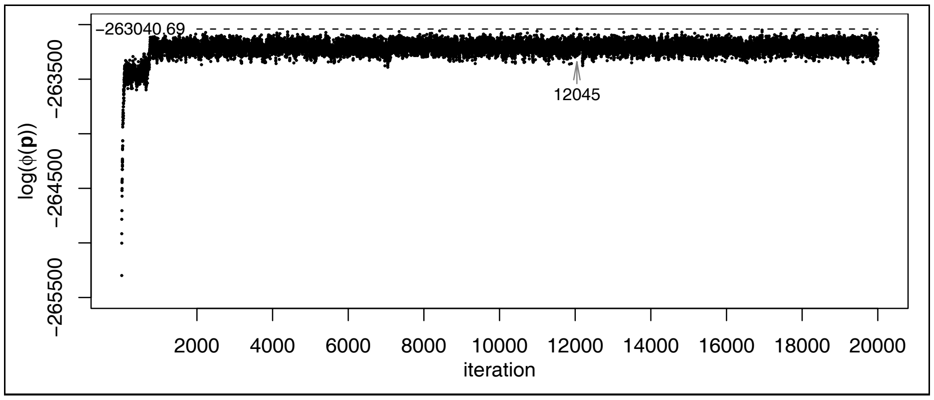
for $S = 72$ and $K = 3$.

We fit the Leukaemia data to the cluster models. We plot the log posterior probability $\log(\phi(\mathbf{p}))$ against the number of clusters of the Bayesian hierarchical clustering procedures (BH) in Figure 5, and against sweep number for 20000 sweeps of the stochastic search (SS) in Figure 6. In the plots, we also indicate where the best partitions are. In this case, stochastic search (SS) performs much better than Bayesian hierarchical clustering procedures (BH) in terms of quality, in that it discovers higher posterior probability partitions.

4 Optimal Bayesian clustering using a pairwise coincidence loss function

As previously commented, the maximum a posteriori partition has no objective status as a best estimate of the clustering of the data. Posterior modes can be increasingly unrepresentative measures of the 'centre' of a posterior distribution as the dimensionality of the unknown parameter increases. In normative Bayesian theory, point estimation can only be accomplished after specifying a loss function: the optimal estimate is that parameter value minimising the posterior expected loss. That is, if the focus is on the partition $\mathbf{p}$, we have to

Figure 6: Log posterior partition probability vs sweep number, using SS algorithm applied to the Leukaemia data.



define a loss function $L(\mathbf{p}, \widehat{\mathbf{p}})$, the cost incurred in declaring that the true partition is $\widehat{\mathbf{p}}$ when it is really $\mathbf{p}$, and choose as our optimal partition that $\widehat{\mathbf{p}}$ which minimises $E(L(\mathbf{p}, \widehat{\mathbf{p}})|\mathbf{y})$. Of course, the partition which maximises the posterior probability is the optimal Bayesian estimator under zero-one loss; however, we regard this as very weak justification for using it as we do not find this loss function intuitively appealing.

The requirements of exchangeability of subset labels and items imposes strong constraints on the form of $L(\cdot, \cdot)$, and the interests of tractability are even more demanding. We choose to use one of the loss functions discussed by Binder (1978, 1981), which considers pairs of items and incurs a penalty for each pair that are clustered together when they should not be, and vice versa. That is, if c_i denotes the allocation variable: $c_i = k$ if and only if $i \in C_k$, we define

$$L(\mathbf{p}, \widehat{\mathbf{p}}) = \sum_{(i,j)\in\mathcal{M}} \left(a\mathbb{I}[c_i = c_j, \widehat{c}_i \neq \widehat{c}_j] + b\mathbb{I}[c_i \neq c_j, \widehat{c}_i = \widehat{c}_j] \right).$$

Here $\mathcal{M} = \{(i, j) : i < j; i, j \in \{1, \ldots, n\}\}$, and a and b are given non-negative constants, the unit costs for each kind of pairwise misclassification.

The posterior expected loss is evidently

$$E(L(\mathbf{p}, \widehat{\mathbf{p}})|\mathbf{y}) = \sum_{(i,j)\in\mathcal{M}} \left(a\mathbb{I}[\widehat{c}_i \neq \widehat{c}_j] P\{c_i = c_j|\mathbf{y}\} + b\mathbb{I}[\widehat{c}_i = \widehat{c}_j] P\{c_i \neq c_j|\mathbf{y}\} \right)$$

Let us abbreviate the *posterior coincidence probabilities* $P\{c_i = c_j|\mathbf{y}\} = \rho_{ij}$, then we can write

$$E(L(\mathbf{p}, \widehat{\mathbf{p}})|\mathbf{y}) = a \sum_{(i,j)\in\mathcal{M}} \rho_{ij} - (a + b) \sum_{(i,j)\in\mathcal{M}} \mathbb{I}[\widehat{c}_i = \widehat{c}_j](\rho_{ij} - K)$$

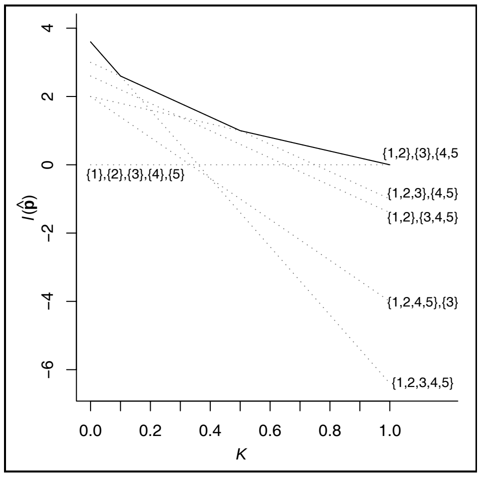
where $K = b/(a + b) \in [0, 1]$. Minimising the posterior expected loss is thus equivalent to maximising

$$\ell(\widehat{\mathbf{p}}, K) = \sum_{(i,j)\in\mathcal{M}} \mathbb{I}[\widehat{c}_i = \widehat{c}_j](\rho_{ij} - K) \tag{14}$$

over choice of $\widehat{\mathbf{p}}$.

Curves of the objective function $\ell(\widehat{\mathbf{p}}, K)$ regarded as a function of K are linear and have non-positive integer slopes and non-negative intercepts, for each $\widehat{\mathbf{p}}$. These functions of K characterise the quality of each possible $\widehat{\mathbf{p}}$, and the whole ensemble of such functions determines in particular for which K, if any, each partition is optimal, as well as defining the optimal $\widehat{\mathbf{p}}$ for each K. Our approach, therefore, is to consider all values of K

Figure 7: The objective function $\ell(\hat{\mathbf{p}}, K)$ vs K for various partitions, for the illustrative Example.



simultaneously.

Example *Suppose there are 5 items and that the partitions of these have probabilities*

$$\begin{array}{ll} \mathbf{p}_1 = \{\{1,2,3\},\{4,5\}\} & \mathbb{P}\{\mathbf{p} = \mathbf{p}_1\} = 0.5 \\ \mathbf{p}_2 = \{\{1,2\},\{3\},\{4,5\}\} & \mathbb{P}\{\mathbf{p} = \mathbf{p}_2\} = 0.2 \\ \mathbf{p}_3 = \{\{1,2\},\{3,4,5\}\} & \mathbb{P}\{\mathbf{p} = \mathbf{p}_3\} = 0.3 \end{array}$$

The ρ matrix can be calculated easily based on those probabilities.

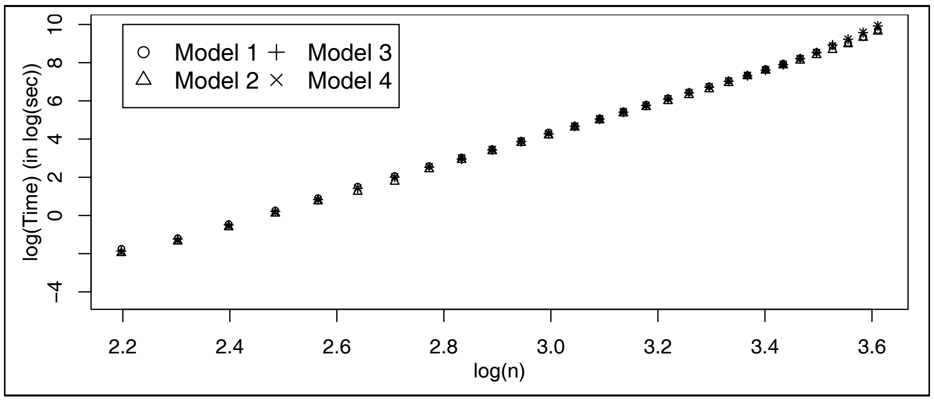
	1	2	3	4	5
1	–	1	0.5	0	0
2	–	–	0.5	0	0
3	–	–	–	0.3	0.3
4	–	–	–	–	1
5	–	–	–	–	–

Figure 7 shows the objective function $\ell(\widehat{\mathbf{p}}, K)$ and related partitions. The dotted lines represent the performance of various partitions. The solid curves represent the optimal performance for each value of K. It is easy to observe that the solid curve is formed from three straight line segments. Each of these segments corresponds to a partition that is optimal for certain K; note that one of these three has zero posterior probability, while one of those with positive posterior probability is never optimal.

4.1 A binary integer programming formulation

We will represent the maximisation of (14) as a binary integer programming problem. Let us replace the indicator function $\mathbb{I}[\hat{c}_i = \hat{c}_j]$ by a binary (0/1) variable X_{ij}; the objective function is a linear combination of the X_{ij} with weights $(\rho_{ij} - K)$. Since $\hat{c}$ represents a proper partition of the items, the X_{ij} are subject to constraints. It is easy to see that what we require is that for all triples (i, j, k), if $X_{ij} = 1$ then $X_{ik} = X_{jk}$. These boolean constraints can be represented as algebraic inequalities, and we find that maximisation of (14) is equivalent to

Figure 8: Running time vs problem size for simulated mixture models 1–4, using linear programming.



solving the optimisation problem:

$$\begin{aligned} &\max_{\{X_{ij}:i,j\in\mathcal{M}\}} \sum_{i,j\in\mathcal{M}} X_{ij}\left(\rho_{ij}-K\right) \\ &\text{s.t. } X_{ij}+X_{ik}-X_{jk}\leq 1 \text{ for } \{i\neq j\neq k\neq i : i,j,k\in\{1,\ldots,n\}\} \\ &\quad X_{ij}\in\{0,1\} \text{ for } i,j\in\mathcal{M} \end{aligned} \tag{15}$$

We conduct a modest experiment to study the feasability of solving this programming problem, although Bansal et al. (2004) have already shown that this is an NP hard problem. We use data sets of n points from each of models 1–4, for various values of n, each replicated 10 times. The regression marginal (5) is fitted to each data set, using a Dirichlet process prior with settings $\theta = 1.0, a_0 = 1.0, b_0 = 0.01, m_0 = \mathbf{0}, t_0 = 0.01\mathbf{I}$. The matrix $[\mathbf{x}_1 \cdots \mathbf{x}_S]'$ is chosen to be an identity matrix $\mathbf{I}_{S\times S}$. The MCMC sampler discussed in Section 3.1 was used to generate 10,000 sweeps following 10,000 burn-in for each data sequence. We then estimated ρ_{ij} for each data sequence using the last 10,000 sweeps. We use the **lpSolve** package (Berkelaar et al, 2006) of the statistical system **R** to solve the programming problem (15) for each $K \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. The code in this package solves general mixed linear-integer programmes. In all of the computations using **lpSolve** reported in this paper, exactly the same solutions are obtained whether or not the solution is constrained to be integer; however, it is possible to construct sets of ρ_{ij} for which this is not true. The times spent on solving the problem are recorded in seconds. In Figure 8 we plot $\log(\text{Time})$ against $\log(n)$ for each data set and least square regression parameters are estimated. In our very rough prediction, we may spend 225.57 billion years to solve a programming program with $n = 1000$. This demonstrates the need for a faster algorithm to perform the optimisation, or at least to obtain an approximation to the optimum.

4.2 A novel algorithm

When $K = 1$ or $K = 0$, the optimal partitions are trivially obtained – they are those corresponding to $X_{ij} = 0$ and $X_{ij} = 1$ for all i, j in the set $\mathcal{M}$, respectively. For other K, as we have seen, the solution is not obvious and the problem NP hard. We propose an iterative procedure that gives a locally optimal solution for the problem. The main idea is to solve the problem partially in each iteration; see the following algorithm 1. We subsequently introduce a procedure that gives locally optimal solutions simultaneously for all $K \in [0, 1]$; see Algorithm 2.
Algorithm 1 *Define $\mathcal{M}_i = \{(1,i),\ldots,(i-1,i),(i,i+1),\ldots,(i,n)\}$. Given a value K and a partition* $\mathbf{p}$*,*

Step 1 For $i = 1, \dots, n$, given $\{X_{jk} : j, k \in \mathcal{M} \backslash \mathcal{M}_i\}$ are fixed, solve a binary integer programming problem

$$\max_{\{X_{jk}:j,k\in\mathcal{M}_i\}} \sum_{j,k\in\mathcal{M}_i} X_{jk}\,(\rho_{jk} - K)$$

$$s.t.\ X_{ij} + X_{ik} - X_{jk} \le 1 \ for\ \{i \ne j \ne k \ne i : i, j, k \in \{1, \dots, n\}\}$$

$$X_{ij} \in \{0, 1\}\ for\ i, j \in \mathcal{M}$$

Step 2 Go to step 1 if the value of the objective function evaluated at the new partition exceeds that on the previous iteration.

Algorithm 1 expresses the idea that we can seek to increase the objective function by cycling over items i, taking i out of the current partition and reallocating it in the optimal way: either by assigning it to an existing cluster or by creating a new one.

As we vary K and thus the relative costs of the two kinds of pairwise error in clustering, the optimal partition varies. If K is not pre-determined, we may repeat Algorithm 1 for several choices of K, but there are advantages of convenience and efficiency in considering all $K \in [0, 1]$ simultaneously. Empirically, we find this also helps to avoid converging to local optima.

As we see from (15), each partition $\mathbf{p}$ is characterised by a non-increasing linear function of K, with intercept and slope depending on $\mathbf{p}$. For any set of partitions $\mathcal{P}$, we can define

$$\ell(\mathcal{P}, K) = \max_{\mathbf{p}\in\mathcal{P}} \ell(\mathbf{p}, K)$$

and by basic properties of convexity, this is a non-increasing convex polygonal (piecewise linear) function of K for each $\mathcal{P}$, and is non-decreasing (with respect to set inclusion) in $\mathcal{P}$ for each K. Further, $\ell(\mathcal{P}, K)$ can be parameterised by a set of increasing values $0 = K_0 < K_1 < \cdots < K_r = 1$ and partitions $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_r \in \mathcal{P}$, with

$$\ell(\mathcal{P}, K) = \ell(\mathbf{p}_s, K) \qquad \text{for all } K \in [K_{s-1}, K_s] \tag{16}$$

for $s = 1, 2, \dots, r$. Note that $\mathbf{p}_s$ is optimal among $\mathbf{p} \in \mathcal{P}$ ($\ell(\mathbf{p}_s, K) \ge \ell(\mathbf{p}, K)$) for all $K \in [K_{s-1}, K_s]$.

Our true objective is, for every K, to maximise $\ell(\mathbf{p}, K)$ over the set $\widetilde{\mathcal{P}}$ of *all* partitions; we could then use (16) with $\mathcal{P} = \widetilde{\mathcal{P}}$ to read off the optimal partition (which is evidently constant on intervals of K) for each K. For any subset $\mathcal{P} \subseteq \widetilde{\mathcal{P}}$, $\ell(\mathcal{P}, K)$ is a lower bound for $\ell(\widetilde{\mathcal{P}}, K)$. Algorithm 2 is a heuristic for iteratively increasing $\mathcal{P}$ and hence the lower bound $\ell(\mathcal{P}, K)$ to obtain successive approximations to the optimum.

Algorithm 2 *Initiate the algorithm by setting $\mathcal{P} = \{\mathbf{p}_a, \mathbf{p}_s\}$ where $\mathbf{p}_a$ is the partition that has all items in one cluster and $\mathbf{p}_s$ is the partition consisting of all singleton clusters.*

Step 1 Take a pair of adjacent partitions, say $\mathbf{p}_s$ and $\mathbf{p}_{s+1}$, from the set representing $\ell(\mathcal{P}, K)$ defined in (16).

Step 2 Run Algorithm 1 twice using the value K_s, starting from partitions $\mathbf{p}_s$ and $\mathbf{p}_{s+1}$. The resulting partitions are $\mathbf{p}^*_s$ and $\mathbf{p}^*_{s+1}$.

Step 3 Insert $\mathbf{p}^*_s$ and $\mathbf{p}^*_{s+1}$ into the set $\mathcal{P}$.

Step 4 Recompute the representation (16) for the modified set of partitions $\mathcal{P}$.

Step 5 End if all pairs $(\mathbf{p}_s, \mathbf{p}_{s+1})$ have been visited and the representation of the set $\mathcal{P}$ is unchanged; go to step 1 otherwise.

We perform an experiment based on the same settings as in the previous section, using Algorithm 2 on the interval $[0, 0.99]$. (We find that the optimal partition changes rapidly for K near 1, so that Algorithm 2 is computationally expensive applied to the full interval.) In Figure 9 we plot the $\log(\text{Time})$ against $\log(n)$ for each data set, and estimate the least squares regression. Interestingly, the run time of the algorithm is an increasing function of the number of hidden components. We predict a run time of 3.51 hours to solve the optimisation for $n = 1000$ items under model 4.

Figure 9: Running time vs problem size for simulated mixture models 1–4, using Algorithm 2.

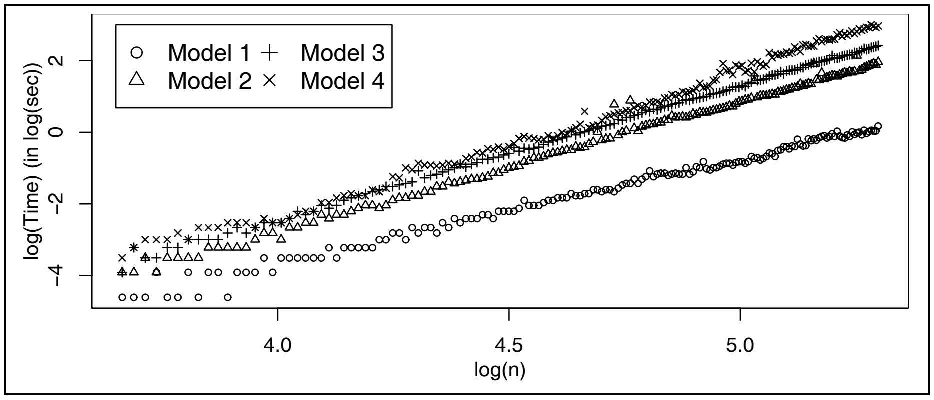


Table 1: Proportion of 10,000 replicates from model 1 for which Algorithm 2 obtains the true optimum, for a range of K and n.

	$K = 0.1$	$K = 0.2$	$K = 0.3$	$K = 0.4$	$K = 0.5$	$K = 0.6$	$K = 0.7$	$K = 0.8$	$K = 0.9$
$n = 3$	1	1	1	1	1	1	1	1	1
$n = 4$	1	1	1	1	1	1	1	1	1
$n = 5$	1	1	1	1	1	1	1	1	0.9999
$n = 6$	0.9990	0.9993	0.9994	0.9998	0.9996	0.9994	0.9996	0.9998	0.9998
$n = 7$	0.9985	0.9983	0.9984	0.9985	0.9984	0.9986	0.9993	0.9993	0.9995
$n = 8$	0.9971	0.9974	0.9982	0.9986	0.9982	0.9980	0.9982	0.9991	0.9991
$n = 9$	0.9966	0.9978	0.9978	0.9966	0.9977	0.9979	0.9981	0.9990	0.9988
$n = 10$	0.9986	0.9980	0.9983	0.9987	0.9983	0.9984	0.9985	0.9991	0.9984
$n = 11$	0.9983	0.9986	0.9987	0.9985	0.9987	0.9982	0.9979	0.9982	0.9982
$n = 12$	0.9987	0.9981	0.9983	0.9978	0.9987	0.9986	0.9993	0.9993	0.9984
$n = 13$	0.9989	0.9987	0.9987	0.9986	0.9990	0.9987	0.9986	0.9991	0.9982
$n = 14$	0.9985	0.9993	0.9996	0.9990	0.9986	0.9989	0.9991	0.9985	0.9987
$n = 15$	0.9992	0.9997	0.9997	0.9992	0.9994	0.9993	0.9997	0.9987	0.9984

As pointed out by a referee, we may also attempt to find the optimal partition using stochastic search: that is, to deliver as an estimate the partition with smallest expected loss encountered during the Monte Carlo run. This is somewhat similar to the approach of Dahl (2006), in a different model, although he does not use an explicit decision theoretic formulation. We have not explored this, but note that this method always yields one of the partitions actually visited during the run, whereas our approach considers a much larger set of possible partitions.

4.3 Numerical experiments

4.3.1 Assessment of heuristic approximation

Our assessment of the performance of the algorithm uses models 1–4, but with small data sets, n varying from 3 to 15. 10000 replications are made of each case. We fit the regression marginal (5) to each data set, with the Dirichlet process prior. The matrix $[\mathbf{x}_1 \cdots \mathbf{x}_S]'$ is chosen to be the identity matrix $\mathbf{I}_{S\times S}$. The sampling uses the Gibbs sampler with parameters $\theta = 1.0, a_0 = 1.0, b_0 = 0.01, m_0 = \mathbf{0}, t_0 = 0.01\mathbf{I}$ and generates 10,000 partitions following 10,000 burn-in for each data set. We estimate ρ_{ij} for each data set based on the last 10,000 sweeps. We use our Algorithm 2 to generate best partitions for $K \in [0, 0.99]$, and compare these to the true optima, calculated using **lpSolve** as in Section 4.1. Tables 1 − 4 show the proportions of runs in which the algorithm finds the true optimum for different K and n, while Tables 5 − 8 show the mean of the ratios of the optimised $\ell(\mathbf{p}, K)$ for different K and n.

Table 2: Proportion of 10,000 replicates from model 2 for which Algorithm 2 obtains the true optimum, for a range of K and n.

	$K = 0.1$	$K = 0.2$	$K = 0.3$	$K = 0.4$	$K = 0.5$	$K = 0.6$	$K = 0.7$	$K = 0.8$	$K = 0.9$
$n = 3$	1	1	1	1	1	1	1	1	1
$n = 4$	1	1	1	1	1	1	1	1	1
$n = 5$	1	1	1	1	1	1	1	1	1
$n = 6$	0.9978	0.9996	0.9999	0.9998	0.9998	0.9998	0.9996	0.9997	0.9995
$n = 7$	0.9871	0.9952	0.9983	0.9995	0.9998	0.9996	0.9997	0.9995	0.9993
$n = 8$	0.9748	0.9875	0.9945	0.9972	0.9989	0.9995	0.9995	0.9993	0.9988
$n = 9$	0.9648	0.9775	0.9904	0.9945	0.9965	0.9979	0.9988	0.9989	0.9980
$n = 10$	0.9564	0.9748	0.9875	0.9927	0.9969	0.9976	0.9983	0.9992	0.9987
$n = 11$	0.9529	0.9696	0.9852	0.9918	0.9955	0.9971	0.9967	0.9975	0.9986
$n = 12$	0.9493	0.9673	0.9829	0.9914	0.9943	0.9955	0.9970	0.9971	0.9983
$n = 13$	0.9478	0.9688	0.9838	0.9899	0.9922	0.9943	0.9950	0.9961	0.9973
$n = 14$	0.9517	0.9702	0.9825	0.9902	0.9926	0.9932	0.9951	0.9965	0.9962
$n = 15$	0.9475	0.9689	0.9836	0.9877	0.9928	0.9937	0.9959	0.9955	0.9968

Table 3: Proportion of 10,000 replicates from model 3 for which Algorithm 2 obtains the true optimum, for a range of K and n.

	$K = 0.1$	$K = 0.2$	$K = 0.3$	$K = 0.4$	$K = 0.5$	$K = 0.6$	$K = 0.7$	$K = 0.8$	$K = 0.9$
$n = 3$	1	1	1	1	1	1	1	1	1
$n = 4$	1	1	1	1	1	1	1	1	1
$n = 5$	1	1	1	1	1	1	1	1	1
$n = 6$	0.9970	0.9993	1	0.9999	0.9999	0.9999	0.9999	0.9997	0.9996
$n = 7$	0.9891	0.9961	0.9991	0.9999	0.9998	0.9998	0.9997	0.9996	0.9992
$n = 8$	0.9752	0.9904	0.9958	0.9978	0.9992	0.9995	0.9992	0.9987	0.9984
$n = 9$	0.9616	0.9817	0.9883	0.9961	0.9982	0.9987	0.9993	0.9989	0.9982
$n = 10$	0.9463	0.9690	0.9828	0.9884	0.9956	0.9975	0.9980	0.9989	0.9987
$n = 11$	0.9377	0.9620	0.9776	0.9862	0.9931	0.9962	0.9979	0.9986	0.9983
$n = 12$	0.9240	0.9582	0.9745	0.9854	0.9907	0.9943	0.9970	0.9986	0.9983
$n = 13$	0.9168	0.9537	0.9741	0.9844	0.9885	0.9919	0.9953	0.9974	0.9974
$n = 14$	0.9164	0.9522	0.9719	0.9828	0.9881	0.9897	0.9926	0.9961	0.9968
$n = 15$	0.9257	0.9517	0.9723	0.9806	0.9867	0.9897	0.9929	0.9945	0.9962

Table 4: Proportion of 10,000 replicates from model 4 for which Algorithm 2 obtains the true optimum, for a range of K and n.

	$K = 0.1$	$K = 0.2$	$K = 0.3$	$K = 0.4$	$K = 0.5$	$K = 0.6$	$K = 0.7$	$K = 0.8$	$K = 0.9$
$n = 3$	1	1	1	1	1	1	1	1	1
$n = 4$	1	1	1	1	1	1	1	1	1
$n = 5$	1	1	1	1	1	1	1	1	1
$n = 6$	0.9985	0.9999	0.9998	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
$n = 7$	0.9925	0.9981	0.9991	0.9996	0.9997	0.9995	0.9994	0.9992	0.9987
$n = 8$	0.9773	0.9917	0.9959	0.9983	0.9991	0.9991	0.9987	0.9977	0.9968
$n = 9$	0.9626	0.9807	0.9894	0.9942	0.9962	0.9981	0.9984	0.9969	0.9955
$n = 10$	0.9581	0.9759	0.9831	0.9893	0.9938	0.9968	0.9978	0.9975	0.9957
$n = 11$	0.9519	0.9698	0.9790	0.9869	0.9905	0.9948	0.9958	0.9984	0.9973
$n = 12$	0.9522	0.9658	0.9755	0.9828	0.9909	0.9934	0.9954	0.9971	0.9961
$n = 13$	0.9527	0.9677	0.9780	0.9821	0.9870	0.9917	0.9939	0.9967	0.9961
$n = 14$	0.9539	0.9676	0.9804	0.9825	0.9860	0.9906	0.9921	0.9958	0.9969
$n = 15$	0.9584	0.9709	0.9782	0.9847	0.9865	0.9888	0.9945	0.9948	0.9953

Table 5: Mean of the ratio of expected posterior loss calculated by Algorithm 2 to the true minimised expectation, over 10,000 replicates from model 1, for various K and n.

	$K = 0.1$	$K = 0.2$	$K = 0.3$	$K = 0.4$	$K = 0.5$	$K = 0.6$	$K = 0.7$	$K = 0.8$	$K = 0.9$
$n = 3$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 4$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 5$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 6$	1.00032	1.00019	1.00010	1.00000	1.00005	1.00005	1.00002	1.00000	1.00000
$n = 7$	1.00036	1.00032	1.00016	1.00013	1.00008	1.00004	1.00004	1.00001	1.00003
$n = 8$	1.00058	1.00042	1.00019	1.00012	1.00011	1.00008	1.00005	1.00002	1.00001
$n = 9$	1.00084	1.00030	1.00017	1.00017	1.00014	1.00008	1.00006	1.00002	1.00002
$n = 10$	1.00057	1.00032	1.00010	1.00006	1.00007	1.00004	1.00003	1.00002	1.00001
$n = 11$	1.00041	1.00022	1.00007	1.00005	1.00005	1.00006	1.00004	1.00002	1.00002
$n = 12$	1.00043	1.00019	1.00010	1.00007	1.00005	1.00005	1.00001	1.00001	1.00001
$n = 13$	1.00015	1.00014	1.00008	1.00007	1.00004	1.00004	1.00002	1.00001	1.00002
$n = 14$	1.00022	1.00004	1.00002	1.00005	1.00004	1.00004	1.00001	1.00001	1.00001
$n = 15$	1.00010	1.00002	1.00002	1.00003	1.00001	1.00002	1.00000	1.00001	1.00001

Table 6: Mean of the ratio of expected posterior loss calculated by Algorithm 2 to the true minimised expectation, over 10,000 replicates from model 2, for various K and n.

	$K = 0.1$	$K = 0.2$	$K = 0.3$	$K = 0.4$	$K = 0.5$	$K = 0.6$	$K = 0.7$	$K = 0.8$	$K = 0.9$
$n = 3$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 4$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 5$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 6$	1.00063	1.00004	1.00001	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 7$	1.00508	1.00064	1.00009	1.00003	1.00001	1.00002	1.00001	1.00004	1.00006
$n = 8$	1.01201	1.00305	1.00083	1.00023	1.00005	1.00001	1.00001	1.00002	1.00008
$n = 9$	1.01520	1.00436	1.00129	1.00052	1.00016	1.00006	1.00002	1.00005	1.00011
$n = 10$	1.01813	1.00441	1.00121	1.00036	1.00011	1.00003	1.00005	1.00003	1.00006
$n = 11$	1.01723	1.00408	1.00107	1.00037	1.00012	1.00009	1.00006	1.00005	1.00004
$n = 12$	1.01816	1.00469	1.00111	1.00038	1.00020	1.00010	1.00006	1.00005	1.00003
$n = 13$	1.01764	1.00432	1.00116	1.00047	1.00026	1.00012	1.00008	1.00005	1.00004
$n = 14$	1.01568	1.00391	1.00123	1.00048	1.00022	1.00011	1.00007	1.00004	1.00005
$n = 15$	1.01733	1.00420	1.00100	1.00044	1.00018	1.00012	1.00006	1.00004	1.00003

Table 7: Mean of the ratio of expected posterior loss calculated by Algorithm 2 to the true minimised expectation, over 10,000 replicates from model 3, for various K and n.

	$K = 0.1$	$K = 0.2$	$K = 0.3$	$K = 0.4$	$K = 0.5$	$K = 0.6$	$K = 0.7$	$K = 0.8$	$K = 0.9$
$n = 3$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 4$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 5$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 6$	1.00101	1.00017	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 7$	1.00308	1.00037	1.00006	1.00000	1.00000	1.00000	1.00001	1.00004	1.00009
$n = 8$	1.00966	1.00263	1.00069	1.00015	1.00003	1.00003	1.00009	1.00014	1.00019
$n = 9$	1.01412	1.00413	1.00130	1.00023	1.00005	1.00004	1.00003	1.00005	1.00010
$n = 10$	1.01809	1.00552	1.00191	1.00061	1.00011	1.00005	1.00002	1.00003	1.00003
$n = 11$	1.01780	1.00523	1.00177	1.00065	1.00015	1.00006	1.00003	1.00002	1.00002
$n = 12$	1.01822	1.00463	1.00160	1.00059	1.00023	1.00011	1.00008	1.00004	1.00004
$n = 13$	1.02111	1.00538	1.00169	1.00066	1.00024	1.00012	1.00006	1.00003	1.00003
$n = 14$	1.01828	1.00463	1.00144	1.00054	1.00024	1.00016	1.00013	1.00006	1.00004
$n = 15$	1.01742	1.00511	1.00149	1.00060	1.00028	1.00013	1.00008	1.00007	1.00005

Table 8: Mean of the ratio of expected posterior loss calculated by Algorithm 2 to the true minimised expectation, over 10,000 replicates from model 4, for various K and n.

	$K = 0.1$	$K = 0.2$	$K = 0.3$	$K = 0.4$	$K = 0.5$	$K = 0.6$	$K = 0.7$	$K = 0.8$	$K = 0.9$
$n = 3$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 4$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 5$	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 6$	1.00025	1.00000	1.00001	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
$n = 7$	1.00104	1.00013	1.00003	1.00001	1.00000	1.00000	1.00000	1.00000	1.00007
$n = 8$	1.00455	1.00119	1.00037	1.00007	1.00002	1.00003	1.00011	1.00020	1.00015
$n = 9$	1.00786	1.00243	1.00088	1.00023	1.00006	1.00004	1.00006	1.00011	1.00014
$n = 10$	1.00914	1.00312	1.00112	1.00037	1.00013	1.00005	1.00005	1.00013	1.00016
$n = 11$	1.01060	1.00332	1.00128	1.00048	1.00020	1.00008	1.00005	1.00005	1.00008
$n = 12$	1.00960	1.00325	1.00132	1.00049	1.00018	1.00011	1.00008	1.00005	1.00008
$n = 13$	1.00849	1.00257	1.00092	1.00046	1.00026	1.00013	1.00005	1.00003	1.00006
$n = 14$	1.00767	1.00268	1.00095	1.00049	1.00024	1.00011	1.00008	1.00003	1.00002
$n = 15$	1.00708	1.00201	1.00084	1.00035	1.00019	1.00013	1.00004	1.00004	1.00004

4.3.2 Optimal vs. maximum probability partitions

To get some appreciation of the difference between (approximate) MAP partitions and (approximate) optimal partitions according to our loss function, we make some comparisons of point estimates based on three different approaches, stochastic search (SS), Bayesian hierarchical clustering procedure (BH) and the loss function (Algorithm 2), using simulated data from models 1–4. Similarly to the simulation study in section 3.3.1, we take 100 sets of data from each of the models 1–4, each with $n = 100$ data points. We fit the model (5) with the Dirichlet process prior. The parameters are assigned to be $\theta = 1, a_0 = 1, b_0 = 0.01, \mathbf{m}_0 = [0 \cdots 0]', \mathbf{t}_0 = 0.01\mathbf{I}$.

For the stochastic search method, we run the Gibbs sampler for 20000 sweeps. We record the partition of greatest posterior probability among all 20000 sweeps. We then use Algorithm 2 based on the last 10,000 MCMC samples, following 10,000 burn-in to produce the ρ matrix. We run Algorithm 2 and the ℓ functions (defined by equation (14)) to get the point estimates for $K \in \{0.1, 0.2, \ldots, 0.9\}$. Figure 10 compares log posterior probabilities $\log(\phi(\mathbf{p}))$ for the partitions produced by the three procedures, stochastic search (SS), Bayesian hierarchical clustering procedure (BH) and the loss function method (Algorithm 2). As expected, the results show that the approximate MAP methods tend to yield higher probability partitions, with SS outperforming BH on average. However, to an extent increasing with the number of components in the simulation of the data, the loss function approach nevertheless produces higher probability partitions for a substantial proportion of the data sets.

On the other hand, Figure 11 compares the quantity $\ell(\mathbf{p}, K)$ for the partitions produced by the three procedures. The results show that the loss function nethod (Algorithm 2) always yields partitions with lower posterior expected loss than the approximate MAP methods, particularly for larger values of K. Among the methods inferior by this criterion, SS is again slightly superior to BH, although for smaller values of K, all three methods perform very similarly.

4.3.3 Optimal partition of Galaxy and Leukaemia data sets

Now we consider the Galaxy data. We calculate the ρ matrix based on the last 10000 partitions sampled. We run Algorithm 2 and the ℓ functions (defined by equation (14)) for the partitions in $\mathcal{P}$ are plotted in Figure 12. Here we can see that we have the same best partition for all $K \in [0.1, 0.9]$. Moreover, this partition is the same as that found by SS and BH, and we conclude that for this data set, the optimal partition for such values of K coincides with the maximum a posteriori partition.

Finally we present the case of the Leukaemia data. Figure 13 shows $\ell(\mathcal{P}, K)$ over the interval [0,1]. We also plot the ℓ functions for the partitions produced by the SS and BH procedures. For this loss function, these approximate maximum a posteriori partitions are not optimal whatever the value of K, although the result from the BH procedure is close to optimal for $K \approx 0.08$. We calculate $\log(\phi(\mathbf{p}))$ for the partitions generated by the algorithm for different Ks, and these, together with the log posterior probabilities for the best partitions found by the SS and BH algorithms, are displayed in Table 9. It is interesting to note that in this example, for $K = 0.1, 0.2, 0.3$, the highest posterior probability partitions are produced by our algorithm, which thus produces the best partitions under both the MAP and loss function criteria! Using the same computer resources as in the experiment in the last section, the computer program takes approximately 21 mins and 7292 mins for the Galaxy data and the Leukaemia data respectively.

5 Conclusion

Clustering is an important and challenging problem, with implications in many fields, including biology, engineering and management. The highly complex structure of partition problems makes objective inference difficult, quite apart from the heavy computational burden entailed in any approach when the number of items to be clustered is large. A Bayesian formulation is attractive in principle, but can be particularly demanding

Figure 10: Pairwise comparisons of maximised log partition posterior probability, computed using the SS, BH and Loss function approaches, under simulation models 1–4. In each of the 4 panels, the comparisons are: top left – SS vs loss function; top right – SS vs BH; bottom left – BH vs loss function.

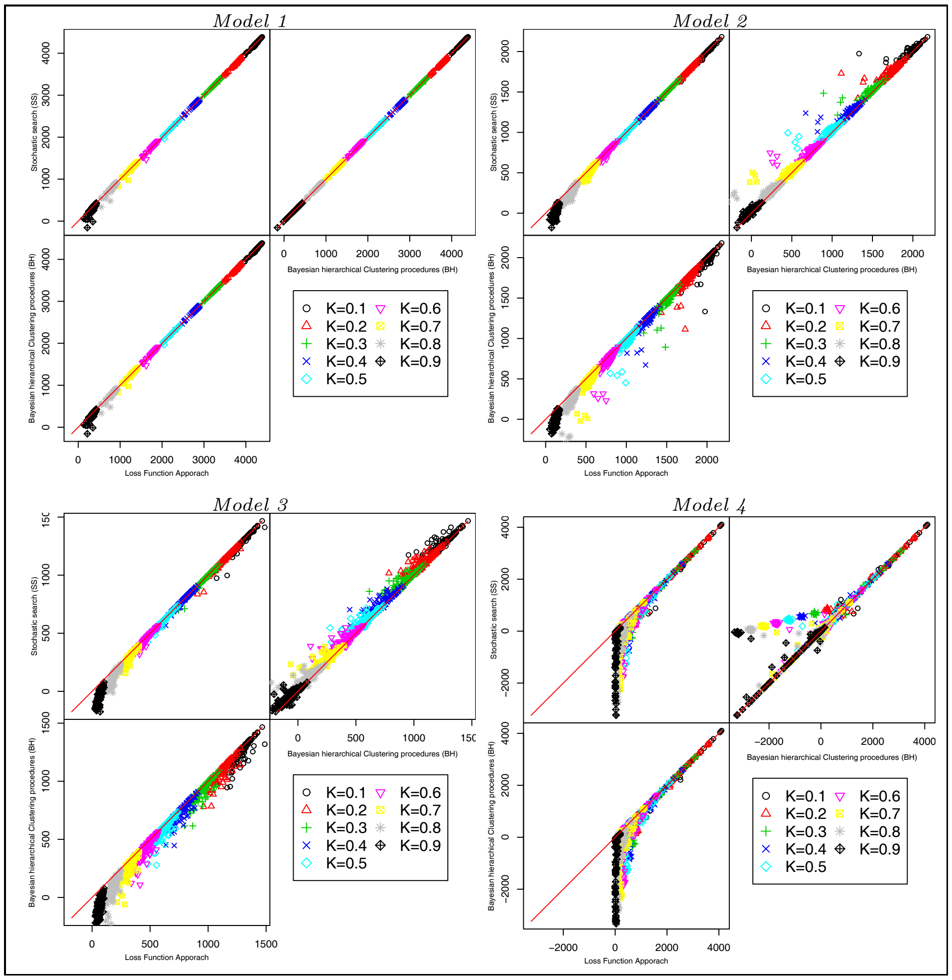


Table 9: Log posterior probabilities for partitions computed by Algorithm 2, and by the SS and BH algorithms.

	$\log(\phi(\mathbf{p}))$
$K = 0.1$	-262774.69
0.2	-262744.51
0.3	-262753.13
0.4	-263451.11
0.5	-265768.95
0.6	-270588.76
0.7	-274760.07
0.8	-279583.08
0.9	-286715.36
SS	-263040.69
BH	-263793.14

Figure 11: Pairwise comparisons of maximised objective function $\ell(\mathbf{p}, K)$, computed using the SS, BH and Loss function approaches, under simulation models 1–4. In each of the 4 panels, the comparisons are: top left – SS vs loss function; top right – SS vs BH; bottom left – BH vs loss function.

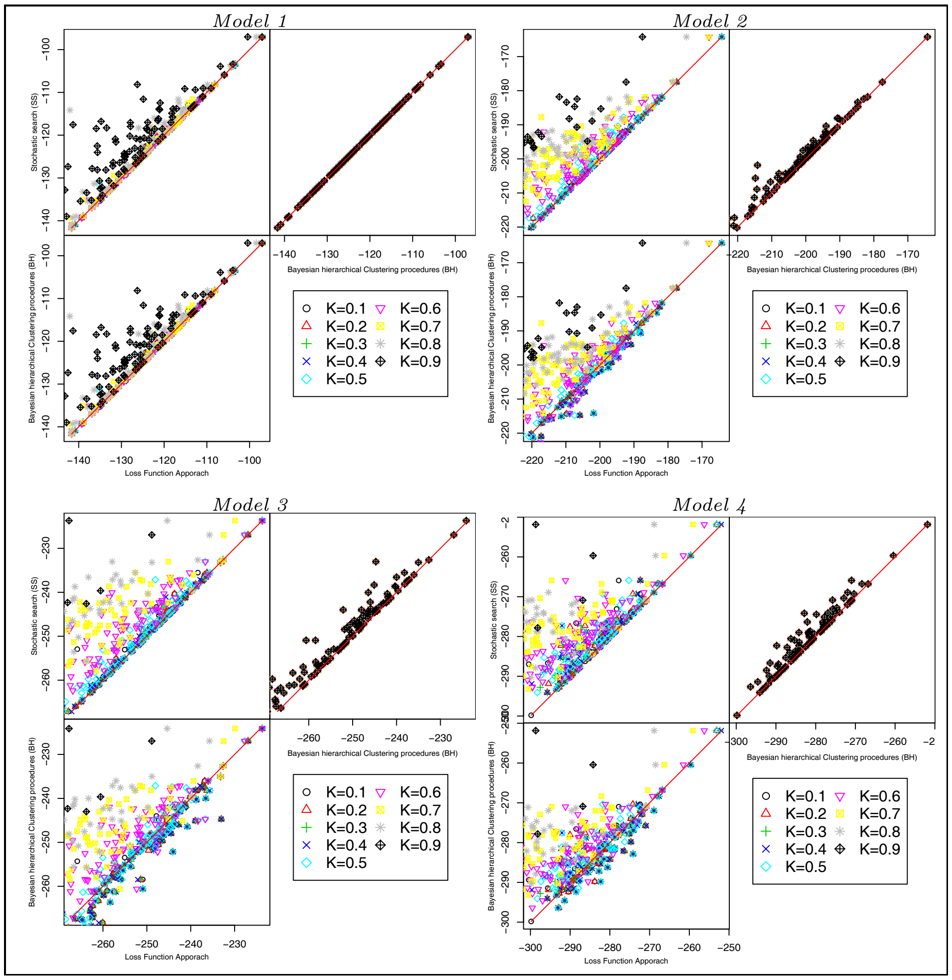


Figure 12: The objective function $\ell(\hat{\mathbf{p}}, K)$ vs K for the partitions considered by Algorithm 2, for the Galaxy data.

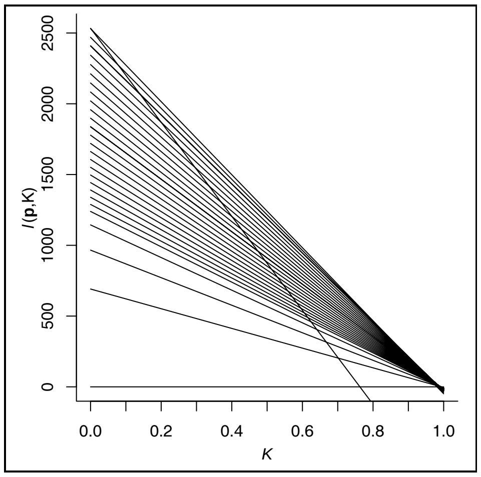
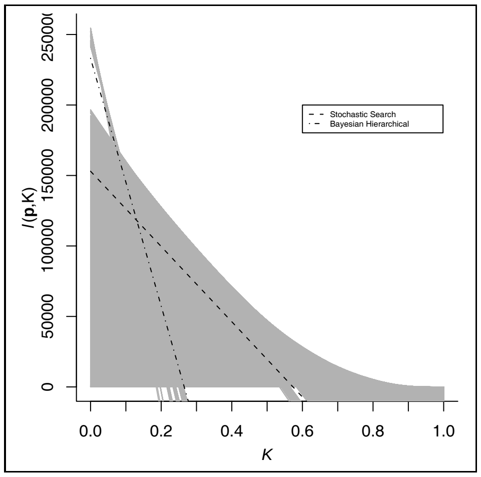


Figure 13: The objective function $\ell(\hat{\mathbf{p}}, K)$ vs K for the partitions considered by Algorithm 2, for the Leukaemia data.



computationally. However, the speed of modern processors is such that it is now practical to follow formal decision theoretic principles in partitioning a few thousand genes in Bayesian analysis of gene expression profiles. Computer time remains an issue, however. Further research should be conducted towards developing better algorithms to reduce the running time.

6 Acknowledgements

This work was carried out with the financial support of the BBSRC Exploiting Genomics initiative.

7 Appendix

7.1 Proof of equation (11)

We first consider the ratio used to combine two clusters, say C_{j_1} and C_{j_2},

$$\frac{m\left(\mathbf{y}_{C_{j_1}\cup C_{j_2}}\right)}{m\left(\mathbf{y}_{C_{j_1}}\right)m\left(\mathbf{y}_{C_{j_2}}\right)} = \exp\left\{-\left[\psi\left(\mathbf{y}_{C_{j_1}\cup C_{j_2}}\right) - \psi\left(\mathbf{y}_{C_{j_1}}\right) - \psi\left(\mathbf{y}_{C_{j_2}}\right)\right]\right\}$$

and the exponent term is now,

$$\begin{aligned}
& \psi\left(\mathbf{y}_{C_{j_1}\cup C_{j_2}}\right)-\psi\left(\mathbf{y}_{C_{j_1}}\right)-\psi\left(\mathbf{y}_{C_{j_2}}\right) \\
= & \sum_{i\in C_{j_1}\cup C_{j_2}}^{n}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)-\sum_{i\in C_{j_1}}^{n}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}}\right)-\sum_{i\in C_{j_2}}^{n}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_2}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_2}}\right) \\
= & \sum_{i\in C_{j_1}}^{n}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)-\sum_{i\in C_{j_1}}^{n}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}}\right) \\
& +\sum_{i\in C_{j_2}}^{n}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)-\sum_{i\in C_{j_2}}^{n}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_2}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_2}}\right) \\
= & e_{j_1}\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)'\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)+e_{j_2}\left(\bar{\mathbf{y}}_{C_{j_2}}-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)'\left(\bar{\mathbf{y}}_{C_{j_2}}-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right) \\
= & e_{j_1}\bar{\mathbf{y}}'_{C_{j_1}}\bar{\mathbf{y}}_{C_{j_1}}+e_{j_2}\bar{\mathbf{y}}'_{C_{j_2}}\bar{\mathbf{y}}_{C_{j_2}}-\left(e_{j_1}+e_{j_2}\right)\bar{\mathbf{y}}'_{C_{j_1}\cup C_{j_2}}\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}} \\
= & \frac{e_{j_1}e_{j_2}}{e_{j_1}+e_{j_2}}\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}\right)'\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}\right).
\end{aligned}$$

This equality will be useful:

$$\begin{aligned}
& \sum_{i\in C_{j_1}\cup C_{j_2}}^{n}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right) \\
& \quad -\sum_{i\in C_{j_1}}^{n}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}}\right)-\sum_{i\in C_{j_2}}^{n}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_2}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_2}}\right) \\
& \qquad =\frac{e_{j_1}e_{j_2}}{e_{j_1}+e_{j_2}}\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}\right)'\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}\right).
\end{aligned} \tag{17}$$

7.2 Proof of equation (12)

Similarly to the proof of (11), we need only consider the exponent term. In this case we first prove an equality

$$\begin{aligned}
& \sum_{i_1\in C_{j_1}}\sum_{i_2\in C_{j_2}}\left(\mathbf{y}_{i_1}-\mathbf{y}_{i_2}\right)'\left(\mathbf{y}_{i_1}-\mathbf{y}_{i_2}\right) \\
= & \sum_{i_1\in C_{j_1}}\sum_{i_2\in C_{j_2}}\left(\mathbf{y}_{i_1}-\bar{\mathbf{y}}_{C_{j_1}}+\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}+\bar{\mathbf{y}}_{C_{j_2}}-\mathbf{y}_{i_2}\right)'\left(\mathbf{y}_{i_1}-\bar{\mathbf{y}}_{C_{j_1}}+\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}+\bar{\mathbf{y}}_{C_{j_2}}-\mathbf{y}_{i_2}\right) \\
= & e_{j_2}\sum_{i_1\in C_{j_1}}\left(\mathbf{y}_{i_1}-\bar{\mathbf{y}}_{C_{j_1}}\right)'\left(\mathbf{y}_{i_1}-\bar{\mathbf{y}}_{C_{j_1}}\right)+e_{j_1}e_{j_2}\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}\right)'\left(\bar{\mathbf{y}}_{C_{j_1}}-\bar{\mathbf{y}}_{C_{j_2}}\right) \\
& \quad +e_{j_1}\sum_{i_2\in C_{j_2}}\left(\bar{\mathbf{y}}_{C_{j_2}}-\mathbf{y}_{i_2}\right)'\left(\bar{\mathbf{y}}_{C_{j_2}}-\mathbf{y}_{i_2}\right).
\end{aligned}$$

This is combined with the equality (17):

$$\begin{aligned}
& \psi\left(\mathbf{y}_{C_{j_1}\cup C_{j_2}}\right)-\psi\left(\mathbf{y}_{C_{j_1}}\right)-\psi\left(\mathbf{y}_{C_{j_2}}\right) \\
= & \left(e_{j_1}+e_{j_2}\right)\sum_{i\in C_{j_1}\cup C_{j_2}}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}\cup C_{j_2}}\right) \\
& \quad -e_{j_1}\sum_{i\in C_{j_1}}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_1}}\right)-e_{j_2}\sum_{i\in C_{j_2}}\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_2}}\right)'\left(\mathbf{y}_i-\bar{\mathbf{y}}_{C_{j_2}}\right) \\
= & \sum_{i_1\in C_{j_1}}\sum_{i_2\in C_{j_2}}\left(\mathbf{y}_{i_1}-\mathbf{y}_{i_2}\right)'\left(\mathbf{y}_{i_1}-\mathbf{y}_{i_2}\right) \\
= & \sum_{i_1\in C_{j_1}}\sum_{i_2\in C_{j_2}}d_{i_1,i_2}.
\end{aligned}$$

7.3 Proof of equation (13)

Again we consider only the exponent term. We first prove

$$\begin{aligned}
& \psi\left(\mathbf{y}_{C_{j_1}\cup C_{j_2}}\right) - \psi\left(\mathbf{y}_{C_{j_1}}\right) - \psi\left(\mathbf{y}_{C_{j_2}}\right) \\
= & \sum_{i\in C_{j_1}\cup C_{j_2}} \left(\mathbf{y}_i - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_1}\cup C_{j_2}}\right)'\left(\mathbf{y}_i - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_1}\cup C_{j_2}}\right) \\
& \quad - \sum_{i\in C_{j_1}} \left(\mathbf{y}_i - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_1}}\right)'\left(\mathbf{y}_i - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_1}}\right) \\
& \quad - \sum_{i\in C_{j_2}} \left(\mathbf{y}_i - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_2}}\right)'\left(\mathbf{y}_i - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_2}}\right) \\
= & \sum_{i\in C_{j_1}} \left([\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_1}} - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_1}\cup C_{j_2}}\right)'\left([\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_1}} - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_1}\cup C_{j_2}}\right) \\
& \quad + \sum_{i\in C_{j_2}} \left([\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_2}} - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_1}\cup C_{j_2}}\right)'\left([\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_2}} - [\mathbf{x}_1\cdots\mathbf{x}_S]'\hat{\boldsymbol{\beta}}_{C_{j_1}\cup C_{j_2}}\right) \\
= & \hat{\boldsymbol{\beta}}'_{C_{j_1}}\mathbf{X}'_{C_{j_1}}\mathbf{X}_{C_{j_1}}\hat{\boldsymbol{\beta}}_{C_{j_1}} + \hat{\boldsymbol{\beta}}'_{C_{j_2}}\mathbf{X}'_{C_{j_2}}\mathbf{X}_{C_{j_2}}\hat{\boldsymbol{\beta}}_{C_{j_2}} - \hat{\boldsymbol{\beta}}'_{C_{j_1}\cup C_{j_2}}\left(\mathbf{X}'_{C_{j_1}}\mathbf{X}_{C_{j_1}} + \mathbf{X}'_{C_{j_2}}\mathbf{X}_{C_{j_2}}\right)\hat{\boldsymbol{\beta}}_{C_{j_1}\cup C_{j_2}} \\
= & \left(\hat{\boldsymbol{\beta}}_{C_{j_1}} - \hat{\boldsymbol{\beta}}_{C_{j_2}}\right)'\left[\left(\mathbf{X}'_{C_{j_1}}\mathbf{X}_{C_{j_1}}\right)^{-1} + \left(\mathbf{X}'_{C_{j_2}}\mathbf{X}_{C_{j_2}}\right)^{-1}\right]^{-1}\left(\hat{\boldsymbol{\beta}}_{C_{j_1}} - \hat{\boldsymbol{\beta}}_{C_{j_2}}\right).
\end{aligned}$$

Finally, note that

$$\mathbf{X}'_{C_{j_1}}\mathbf{X}_{C_{j_1}} = \sum_{i\in C_{j_1}} [\mathbf{x}_1\cdots\mathbf{x}_S]\,[\mathbf{x}_1\cdots\mathbf{x}_S]' \text{ and } \mathbf{X}'_{C_{j_2}}\mathbf{X}_{C_{j_2}} = \sum_{i\in C_{j_2}} [\mathbf{x}_1\cdots\mathbf{x}_S]\,[\mathbf{x}_1\cdots\mathbf{x}_S]'.$$

8 References

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

Banfield, J. D., and Raftery, A. E. (1993), "Model-based Gaussian and mon-Gaussian clustering", *Biometrics*, **49**, 803–821.

Bansal, N., Blum, A., and Chawla, S. (2004), "Correlation clustering", *Machine Learning*, **56**, 89–113.

Barry, D., and Hartigan, J. A. (1992), "Product partition models for change point problems", *The Annals of Statistics*, **20**, 260–279.

Basu, S. and Chib, S. (2003), "Marginal Likelihood and Bayes Factors for Dirichlet Process Mixture Models", *Journal of the American Statistical Association*, **98**, 224–235

Berkelaar, M. and others (2006), "lpSolve: Interface to Lp_solve v. 5.5 to solve linear/integer programs", R package version 5.5.3.

Binder, D. A. (1978), "Bayesian cluster analysis", *Biometrika*, **65**, 31–38.

– – (1981), "Approximations to Bayesian clustering rules", *Biometrika*, **68**, 275–285.

Blackwell, D., and MacQueen, J. B. (1973), "Ferguson distributions via Pólya Urn Schemes", *The Annals of Statistics*, **1**, 353–355.

Brunner, L. J., and Lo, A. Y. (1999), "Bayesian classifications", *Preprint, University of Toronto, Canada.* Available at `http://www.erin.utoronto.ca/~jbrunner/papers/BayesClass.pdf`

Dahl, D. B. (2006), "Model-Based Clustering for Expression Data via a Dirichlet Process Mixture Model", in *Bayesian Inference for Gene Expression and Proteomics, Kim-Anh Do, Peter Müller, Marina Vannucci (Eds.)*, Cambridge University Press. in press.

de Finetti, B. (1930), “Funzione caratteristica di un fenomeno aleatorio”, *Atti Reale Accademia Nazionale dei Lincei, Mem.* **4**, 86–133.

– – (1974), *Theory of Probability 1.* Wiley.

Diaconis, P., and Freedman, D. (1984), “Partial exchangeability and sufficiency”, in *Statistics: Applications and New Directions, eds. J. K. Ghosh and J. Roy*, pp. 205–236.

– – (1987), “A dozen de Finetti-style results in search of a theory”, *Annales de l'Institut Henri Poincaré (B) Probabilités et Statistiques*, **23**, 397–423.

Dubey, A., Hwang, S., Rangel, C., Rasmussen, C. E., Ghahramani, Z., and Wild, D. L. (2004) “Clustering Protein Sequence and Structure Space with Infinite Gaussian Mixture Models”, *Pacific Symposium in Biocomputing World Scientific Publishing, Singapore*, **9**, 399–410.

Dudoit, S., Fridlyand, J. and Speed, T. P. (2002), “Comparison of discrimination methods for the classification of tumors using gene expression data”, *Journal of the American Statistical Association*, **97**, 77–87.

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

– – (1998), “Computing monparametric hierarchical models”, in *Practical Nonparametric and Semiparametric Bayesian Statistics, eds. D. Dey, P. Müller, and D. Sinha, New York: Springer*, pp. 1–22.

Ferguson, T. S. (1973), “A Bayesian analysis of some nonparametric problems”, *The Annals of Statistics*, **1**, 209–230.

Fraley, C., and Raftery, A. E. (2002), “Model-based clustering, discriminant analysis, and density estimation”, *Journal of the American Statistical Association*, **97**, 611–631.

Gentleman, R. and others (2004), “Bioconductor: Open software development for computational biology and bioinformatics”, *Genome Biology*, **5**, R80. http://genomebiology.com/2004/5/10/R80.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D., and Lander, E. S. (1999), “Molecular classification of cancer: class discovery and class prediction by gene expression monitoring”, *Science*, **286**, 531–537.

Gordon, A. D. (1999), *Classification*, 2nd edition. Chapman & Hall.

Green, P. J. (1995), “Reversible jump Markov chain Monte Carlo computation and Bayesian model determination”, *Biometrika*, **82**, 711–732.

Green, P. J., and Richardson, S. (2001), “Modelling heterogeneity with and without the Dirichlet process”, *Scandinavian Journal of Statistics*, **28**, 355–375.

Hartigan, J. A. (1990), “Partition models”, *Communications in statistics – Simulation and computation*, **19**, 2745–2756.

Heard, N. A., Holmes, C. C., and Stephens, D. A. (2006), “A quantitative study of gene regulation involved in the immune response of Anopheline mosquitoes: an application of Bayesian hierarchical clustering of curves”, *Journal of the American Statistical Association*, *101*, 18–29.

Heller, K.A. and Ghahramani, Z. (2005), “Bayesian Hierarchical Clustering”, *Twenty-second International Conference on Machine Learning (ICML-2005).* Available at `http://learning.eng.cam.ac.uk/zoubin/papers/icml05heller.pdf`

Hewitt, E., and Savage, L. J. (1955), “Symmetric measures on Cartesian products”, *Transactions of the American Mathematical Society*, **80**, 470–501.

Hurn, M., Justel, A., and Robert, C. P. (2003), “Estimating mixtures of regressions”, *Journal of Computational and Graphical Statistics*, **12**, 55–79.

Ishwaran, H., and James, L. F. (2001), "Gibbs sampling methods for stick-breaking priors", *Journal of the American Statistical Association*, **96**, 161–173.

– – (2003a), "Generalized weighted Chinese restaurant processes for species sampling mixture models", *Statistica Sinica* **13** 1211–1235.

– – (2003b), "Some further developments for stick-breaking priors: finite and infinite clustering and classification", *Sankhya Series A*, **65**, 577–592.

James, L. F. (2002), "Poisson process partition calculus with applications to exchangeable models and Bayesian Nonparametrics", Available at `http://arXiv.org/abs/math/0205093`.

– – (2005), "Bayesian Poisson process partition calculus with an application to Bayesian Levy moving averages", *The Annals of Statistics*, **33**, 1771–1799.

James, L. F., Lijoi, A. and Prüenster, I. (2005) "Bayesian inference for classes of normalized random measures", Available at `http://arxiv.org/abs/math.ST/0503394`

Kingman, J. F. C. (1975), "Random discrete distributions", *Journal of the Royal Statistical Society: Series B*, **37**, 1–22.

– – (1993), *Poisson Processes*, Oxford University Press.

Kirkpatrick, S., Gelatt, C. D. and Vecchi, M. P., (1983), "Optimization by Simulated Annealing", *Science*, **220**, 671–680.

Lijoi, A., Mena, R. H., and Prünster, I. (2005), "Hierarchical mixture modeling with normalized inverse-Gaussian priors". *Journal of the American Statistical Association*, **14**, 1278–1291.

Lo, A. Y. (1984), "On a class of Bayesian nonparametric estimates. I. Density estimates", *The Annals of Statistics*, **12**, 351–357.

– – (2005), "Weighted Chinese restaurant processes", *COSMOS*, **1**, 59–63.

Lo, A. Y., Brunner, L. J. and Chan, A. T. (1996), "Weighted Chinese restaurant processes and Bayesian mixture models", *Research Report, Hong Kong University of Science and Technology.* Available at `http://www.erin.utoronto.ca/~jbrunner/papers/wcr96.pdf`

MacEachern, S. N. (1994), "Estimating normal means with a conjugate style Dirichlet process prior", *Communications in statistics – Simulation and computation*, **23**, 727–741.

MacEachern, S. N., and Müller, P. (1998), "Estimating mixture of Dirichlet process models", *Journal of Computational and Graphical Statistics* , **7**, 223–238.

– – (2000), "Efficient MCMC schemes for robust model extensions using encompassing Dirichlet process mixture models", *Robust Bayesian analysis*, 295–315.

Medvedovic, M., and Sivaganesan, S. (2002), "Bayesian infinite mixture model based clustering of gene expression profiles", *Bioformatics*, **18**, 1194–1206.

Medvedovic, M., Yeung, K. Y., and Bumgarner, R. E. (2004), "Bayesian mixture model based clustering of replicated microarray data", *Bioformatics*, **20**, 1222–1232.

Neal, R. M. (2000), "Markov chain sampling methods for Dirichlet process mixture models", *Journal of Computational and Graphical Statistics*, **9**, 249–265.

Nobile, A., and Fearnside, A. (2005), "Bayesian finite mixtures with an unknown number of components: the allocation sampler", *Technical Report 05-4, University of Glasgow.* Available at: `http://www.stats.gla.ac.uk/~agostino/mixalloc.pdf`

Pitman, J. (1995), "Exchangeable and partially exchangeable random partitions", *Probability Theory and Related Fields*, **102**, 145–158

– – (1996), "Some developments of the Blackwell–MacQueen urn scheme", in *Statistics, Probability and Game Theory. Papers in honor of David Blackwell, IMS Lecture Note Series, eds. T. S. Ferguson, J. B. MacQueen, L. S. Shapley*, pp. 245–267.

– – (2003), "Poisson Kingman partitions", in *Science and Statistics: A Festschrift for Terry Speed, IMS Lecture Note Series, eds. D. Goldstein*, pp. 1–34. Available at `http://arxiv.org/abs/math/0210396`

Pitman, J., and Yor, M. (1997), "The two-parameter Poisson–Dirichlet distribution derived from a stable subordinator", *The Annals of Probability*, **25**, pp. 855–900.

Quintana F. A., and Iglesias P. L. (2003), "Bayesian clustering and product partition models", *Journal of the Royal Statistical Society: Series B*, **65**, 557–574.

Ray, S. and Mallick, B. (2006), "Functional clustering by Bayesian wavelet methods", *Journal of the Royal Statistical Society: Series B*, **68**, 305–332.

Richardson, S., and Green, P. J. (1997), "On Bayesian analysis of mixtures with an unknown number of components (with discussion)", *Journal of the Royal Statistical Society: Series B*, **59**, 731–792.

Roeder, K. (1990), "Density estimation with confidence sets exemplified by superclusters and voids in the galaxies", *Journal of the American Statistical Association*, **85**, 617–624.

Rue, H. (1995), "New loss functions in Bayesian imaging", *Journal of the American Statistical Association*, **90**, 900–908.

West, M., Müller, P., and Escobar, M. D. (1994), "Hierarchical priors and mixture models, with applications in regression and density estimation", in *A tribute to D. V. Lindley, eds. A. F. M Smith and P. R. Freeman, New York: Wiley.*

Ward, J. H. (1963), "Hierarchical grouping to optimize an objective function", *Journal of the American Statistical Association*, **58**, 236–244.

Yeung, K. Y., Fraley, C., Murua, A., Raftery, A. E., and Ruzzo, W. L. (2001), "Model-based clustering and data transformations for gene expression data", *Bioinformatics*, **17**, 977–987.