Journal of Statistical Software

February 2005, Volume 13, Issue 2. http://www.jstatsoft.org/

BClass: A Bayesian Approach Based on Mixture Models for Clustering and Classification of Heterogeneous Biological Data

Arturo Medrano-Soto
Centro de Ciencias Genómicas

J. Andrés Christen
Centro de Investigación en Matemáticas

Julio Collado-Vides
Centro de Ciencias Genómicas

Abstract

Based on mixture models, we present a Bayesian method (called **BClass**) to classify biological entities (e.g. genes) when variables of quite heterogeneous nature are analyzed. Various statistical distributions are used to model the continuous/categorical data commonly produced by genetic experiments and large-scale genomic projects. We calculate the posterior probability of each entry to belong to each element (group) in the mixture. In this way, an original set of heterogeneous variables is transformed into a set of purely homogeneous characteristics represented by the probabilities of each entry to belong to the groups. The number of groups in the analysis is controlled dynamically by rendering the groups as 'alive' and 'dormant' depending upon the number of entities classified within them. Using standard Metropolis-Hastings and Gibbs sampling algorithms, we constructed a sampler to approximate posterior moments and grouping probabilities. Since this method does not require the definition of similarity measures, it is especially suitable for data mining and knowledge discovery in biological databases. We applied **BClass** to classify genes in **RegulonDB**, a database specialized in information about the transcriptional regulation of gene expression in the bacterium *Escherichia coli*. The classification obtained is consistent with current knowledge and allowed prediction of missing values for a number of genes.

BClass is object-oriented and fully programmed in Lisp-Stat. The output grouping probabilities are analyzed and interpreted using graphical (dynamically linked plots) and query-based approaches. We discuss the advantages of using Lisp-Stat as a programming language as well as the problems we faced when the data volume increased exponentially due to the ever-growing number of genomic projects.

Keywords: genetic databases, bioinformatics, MCMC, mixture models, clustering, data mining.

1. Introduction

The exponential growth of raw biological information represents an unprecedented challenge for biologists and bioinformaticians. Striking breakthroughs in biotechnology currently allow sequencing an average bacterial genome (the total DNA within an organism) in a matter of days. Since 1995 when the first complete sequence of a free-living bacterium *Haemophilus influenzae* was obtained (Flaischmann and *et al* 1995) to November 2004, there are already more than 200 complete genomes available, sampling the three domains of life, at The National Center for Biotechnology information (NCBI, `http://www.ncbi.nih.gov`), and more than 140 are still under way. Furthermore, high throughput experimental approaches have been developed to measure simultaneously the expression of all genes within an organism, both at the level of messenger RNA (Brown and Botstein 1999) and protein content (Anderson *et al.* 2000; Dutt and Lee 2000); such approaches have led to the emergence of the fields of transcriptomics and proteomics respectively. Statistical interpretation of transcriptome results is not a straightforward endeavor since different growth conditions, different RNA extraction procedures and different microarray-building systems hamper comparisons between experiments. Pitfalls, challenges and limits of these approaches have been adequately discussed elsewhere Danchin and Sekowska (2000). Experiments focused on individual biological systems, which involve a few genes, continue to appear in the literature and a substantial part of the resulting information is available in specialized databases. The inescapable consequence is that the pace at which raw experimental data is generated has greatly exceeded our ability to extract insightful knowledge from such information. For example, it is not trivial to infer from sets of coexpressed genes under a given experimental condition, which genes are corregulated, what are the regulatory proteins that regulate them, and even much more complicated, what is the network of regulatory interactions that produces the observed expression patterns. More robust methods and enhanced computational tools are required to come to grips with this problem by integrated analyses of heterogeneous data types.

We focused on the problem of clustering and knowledge discovery in **RegulonDB**, a biological database specialized in information about operon organization and transcriptional regulation of gene expression in the bacterium *Escherichia coli* (Salgado *et al.* 2004). The specific goal is to build a statistical framework that, hopefully, will allow uncovering previously unknown relationships among genes, given the diverse attributes that describe them. In our strategy we use a classical mixture model to tackle the problem of multivariate, heterogeneous classification and clustering. We call the resulting software **BClass**. Let $\boldsymbol{X} = (\boldsymbol{x}_{iv})$ be a (non-relational) database where $\boldsymbol{x}_{iv}$ represents attribute v of gene (entry) i; $i = 1, 2, \ldots, n$ and $v = 1, 2, \ldots, C$. Let also $\boldsymbol{x}_i$ be the ith row of $\boldsymbol{X}$; all the attributes for gene i. By heterogeneous databases we mean that the $\boldsymbol{x}_{iv}$'s may be very different in nature: continuous, discrete, categorical, etc. See Section 6 for an application example of **BClass** to **RegulonDB**.

A variety of clustering algorithms based on dissimilarity or distance measures are currently available. Methods for clustering and classification of heterogeneous, mixed, variables include converting variables to homogeneous types, analyzing variables separately and finding a weighted average of standardized dissimilarity measures (see, for example, Kaufman and Rousseau 1990; Gordon 1981). Regarding this latter method, many authors have suggested procedures to properly define the weights in a joint dissimilarity measure. However, as yet, there is the caveat that no standard method has been widely accepted and much heuristics is here needed (see Everitt 1993). As an alternative, mixture models have the advantage of

not relying either on distance measures nor on building (or attempting to build) a statistical model of the data analyzed (see Everitt 1993; McLachlan and Basford 1988) The now well known software **AutoClass** of Cheeseman and Stutz (1996) uses mixtures for clustering and classification in complex databases; it is largely based on the techniques presented by Titterington *et al.* (1985). **AutoClass** uses expected maximization (EM) approximations to find, mainly, maximum *a posteriori* (MAP) estimators (point estimators). In this paper, we build a mixture model for $\boldsymbol{X}$ using the assumption of conditional independence across elements in the mixture and attributes; this assumption has been shown both empirically and theoretically to be highly effective (Hand and Keming 2001). We embarked on doing a full Bayesian analysis using Markov Chain Monte Carlo (MCMC, see for example Gamerman 1997) methods to approximate complete posteriors, specifically, all grouping probabilities. This means that, in particular, given the number of groups or components in the mixture (J), we obtain a matrix $\boldsymbol{P} = (p_{ij})$ where p_{ij} is the posterior probability for entry i to belong to group j, $j = 1, 2, \ldots, J$. The whole process may be regarded as a transformation of $\boldsymbol{X}$ to $\boldsymbol{P}$, where now the "attributes" of each entry are its grouping probabilities p_{ij}'s. These new "attributes" are perfectly homogeneous. We then may use, as posterior interpretation tools, simple clustering techniques on $\boldsymbol{P}$ to find clusters. Therefore, both groups and posterior grouping probabilities (p_{ij}) are well defined mathematical objects, and "Clusters" are rather intuitive, and more interpretable, in terms of the database under consideration. Overall, we follow **AutoClass** philosophy of "automatic" classification, that is, setting the least number of parameters and relying on reasonable default values.

The problem of deciding the appropriate number of components in a mixture has been studied by many authors. However, there are only a handful of full Bayesian approaches that state a prior and calculate a posterior for the number of components J in the mixture. The difficulty here is that, as J varies, the dimension of the model changes, and standard MCMC approaches cannot handle chains of variable dimension. Philips and Smith (1996) propose using "jump diffusions", while Richardson and Green (1997) make an application in mixture models of the now popular "Reversible Jump" MCMC, developed by Green (1995). Stephens (1997) proposes yet another approach, based on point-process simulation, to deal with variable number of components. These strategies approach the problem in a full Bayesian setting being applications of general methods for variable dimension models and tend to become quite elaborate. Here, we tailored a procedure that allows handling different number of components, yet keeping J fixed. This is achieved using a vector of indicator ("dimensionality") variables that designate the "alive" and the "dormant" components. Using an alternative prior for the mixing probabilities that prefers parsimonious parameterizations, we construct a Markov Chain Monte Carlo (MCMC) relying on standard Gibbs sampling and Metropolis-Hastings algorithms (see Besag *et al.* 1995). It is worth noting that our method is suited for any set of component distributions, whereas the aforementioned approaches have been explored mainly for Normal mixtures.

The paper is organized as follows. In Section 2 we explain our model and the general form of the hierarchical structure. Specific priors for various distributions of gene attributes are explained in Section 3 and in Section 4 we discuss the posterior analysis of the MCMC output. In Section 5 we describe our general strategy to implement the system in Lisp-Stat. In Section 6 we apply **BClass** to the analysis of **RegulonDB** (Salgado *et al.* 2004) and some comparisons are made with the software **AutoClass**. Finally, a general discussion of the paper is given in Section 7.

2. The model

We use the standard mixture model with allocation variables as explained, for example, in Richardson and Green (1997). However, we add a vector of indicator variables to control the "alive" and the "dormant" groups in the mixture. That is, given a set of (unknown) parameters $\boldsymbol{h}$, the database $\boldsymbol{X}$ has distribution

$$f(\boldsymbol{X} \mid \boldsymbol{h}) = \sum_{j=1}^{J} \frac{\phi_j \pi_j}{\sum_{m=1}^{J} \phi_m \pi_m} f(\boldsymbol{X} \mid \boldsymbol{\theta}_j),$$

where $\pi_j \geq 0$, $\sum_{j=1}^{J} \pi_j = 1$, are the mixing probabilities, $f(\boldsymbol{X} \mid \boldsymbol{\theta}_j)$ are the within-group (component) distributions, and the indicator variables $\phi_j = 0, 1$ are the "dimensionality variables". Thus if $\phi_j = 0$, then group j is dormant (not considered), and if $\phi_j = 1$, group j is alive. We consider the number of components J to be fixed to a suitable large number and by integrating out the ϕ_j's we tackle the problem of "finding" the number of components in the mixture (the components that remained "alive"). We then have that $\boldsymbol{h} = (\boldsymbol{\pi}, \boldsymbol{\phi}, \boldsymbol{\theta})$, where $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_J)$, $\boldsymbol{\phi} = (\phi_1, \phi_2, \ldots, \phi_J)$ and $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_J)$.

We assume that all the component distributions $f(\boldsymbol{x}_i \mid \boldsymbol{\theta}_j)$ belong to the same parametric family. We also assume conditional independence among components and attributes in the following sense,

$$f(\boldsymbol{X} \mid \boldsymbol{\theta}_j) = \prod_{i=1}^{n} f(\boldsymbol{x}_i \mid \boldsymbol{\theta}_j)$$

and

$$f(\boldsymbol{x}_i \mid \boldsymbol{\theta}_j) = \prod_{v=1}^{C} f(\boldsymbol{x}_{iv} \mid \boldsymbol{\theta}_{jv}),$$

where $\boldsymbol{\theta}_{jv}$ are the parameters needed for component j and attribute v. The above assumptions mean that we consider all attributes as conditionally independent *within each group*. This is a fairly reasonable assumption since, once all other sorts of variability are discarded, the internal group variability may be expected to behave as random, non-correlated, error. As supporting evidence, the studies of Hand and Keming (2001) presented an empirical and theoretical performance analysis of the assumption of conditional independence among attributes. They conclude that, although at first glance naive and most likely incorrect,

> The independence Bayes model seems often to perform surprisingly well.

(Hand and Keming 2001, p. 395). Part of the reason is that less parameters are needed to be estimated, thus outperforming models that consider interaction parameters, specially for several attributes. Furthermore, generally speaking, little is known about the databases studied and thus simple models need to be used (this approach is also taken in **AutoClass** and has given promising results, see Cheeseman and Stutz (1996). In order to consider non-independent attributes in the database, we could use multivariate distributions with some correlation structure. The generalization of the techniques developed here to use multivariate attributes is relatively straightforward (see Fraley and Raftery 1998, as an example using normal variables only). However, in this paper we take $f(\boldsymbol{x}_{iv} \mid \boldsymbol{\theta}_{jv})$ to be univariate, thus we write x_{iv}, a scalar. For clarification, say we have $C = 3$ attributes $\boldsymbol{x}_i = (x_{i1}, x_{i2}, x_{i3})$ for each

gene i, and that these are Normal, Multinomial with four levels and Poisson. In such a case we would have $f(\boldsymbol{x}_i \mid \boldsymbol{\theta}_j) = f(x_{i1} \mid \mu_j, \sigma_j) f(x_{i2} \mid p_{j1}, p_{j2}, p_{j3}, p_{j4}) f(x_{i3} \mid \lambda_j)$ with the obvious notation.

Certainly, in general, we do not know which group each gene belongs to. Thus, for gene i, the group this gene belongs to is given by the latent allocation variable $J_i = 1, 2, \ldots, J$. Given these allocation variables, we have

$$f(\boldsymbol{x}_i \mid \boldsymbol{J}, \boldsymbol{\pi}, \boldsymbol{\theta}) = f(\boldsymbol{x}_i \mid \boldsymbol{\theta}_{J_i})$$

where $\boldsymbol{J}$ is the vector of J_i's. That is, given $\boldsymbol{J}$, $\boldsymbol{x}_i$ is drawn from its corresponding group J_i.

For a given J our parameters are $(\boldsymbol{J}, \boldsymbol{\pi}, \boldsymbol{\phi}, \boldsymbol{\theta})$. We assume that $f(\boldsymbol{J}, \boldsymbol{\pi}, \boldsymbol{\phi}, \boldsymbol{\theta}) = f(\boldsymbol{J}, \boldsymbol{\pi}, \boldsymbol{\phi}) f(\boldsymbol{\theta})$ and *a priori*,

$$P(J_i = j \mid \boldsymbol{\pi}, \boldsymbol{\phi}) \propto \phi_j \pi_j$$

with $f(\boldsymbol{J} \mid \boldsymbol{\pi}, \boldsymbol{\phi}) = \prod_{i=1}^{n} f(J_i \mid \boldsymbol{\pi}, \boldsymbol{\phi})$, and we assume that $P[\phi_j = 1] = \alpha$ independently (α will be taken equal to $\frac{1}{2}$, non-informative). Therefore, to construct our prior we are using the decomposition

$$f(\boldsymbol{J}, \boldsymbol{\pi}, \boldsymbol{\phi}, \boldsymbol{\theta}) = f(\boldsymbol{J} \mid \boldsymbol{\pi}, \boldsymbol{\phi}) f(\boldsymbol{\pi}) f(\boldsymbol{\phi}) f(\boldsymbol{\theta}).$$

We take $f(\boldsymbol{\theta}) = \prod_{j=1}^{J} \prod_{v=1}^{C} f(\boldsymbol{\theta}_{jv})$ and the definition of $f(\boldsymbol{\theta}_{jv})$ is left for Section 3. Alternatively, $\boldsymbol{\phi}$ and $\boldsymbol{\pi}$ may be considered not independent *a priori* by, for example, taking $f(\boldsymbol{\phi}, \boldsymbol{\pi}) = f(\boldsymbol{\phi} \mid \boldsymbol{\pi}) f(\boldsymbol{\pi})$, and making ϕ_j to depend on π_j, somehow. However, this will only be relevant for the case when prior information is availbale to distinguish the π_j's, which is not the case for $f(\boldsymbol{\pi})$. An influence diagram (see, for example, Richardson and Green 1997) for our model is presented in Figure 1.

We leave the more mathematical aspects of defining $f(\boldsymbol{\pi})$ and the design of the Markov Chain Monte Carlo algorithm for the Appendix. We now turn to consider the distributions for the attributes in the data base.

3. Distributions for different attributes

In this section we present distributions for specific types of variables that are commonly used in biological databases. However, it should be clear from the type of hierarchical modeling used that any other type of distribution may also be considered, provided a default (proper) prior is stated and sampling from its unknown parameters is properly described. Proper priors are needed since we could encounter empty groups and in such a case we would need to sample from the prior itself, during the MCMC iterations. As far as the MCMC sampler is concerned, no further adjustments are required and the sampling scheme for the rest of the parameters remains the same. Moreover, the prediction section below regarding missing or unobserved values, is completely general and not solely restricted to the distributions presented herein.

3.1. Normal heterocedastic

With respect to Normal variables we have $\boldsymbol{\theta}_{jv} = (\mu_j, \lambda_j)$ and $x_{iv} \mid J_i = j, \boldsymbol{\mu}, \boldsymbol{\lambda} \sim N(\mu_j, \lambda_j)$ where μ_j is the mean and λ_j the precision (inverse of the variance). Let also $\boldsymbol{\mu}$ be the vector of μ_j's and $\boldsymbol{\lambda}$ the vector of λ_j's. Richardson and Green (1997) take $\mu_j \sim N(\mu_0, \lambda_0)$ and $\lambda_j \sim Ga(\alpha, \beta)$ independently for all j and μ_0, λ_0 and α fixed to some data dependent values.

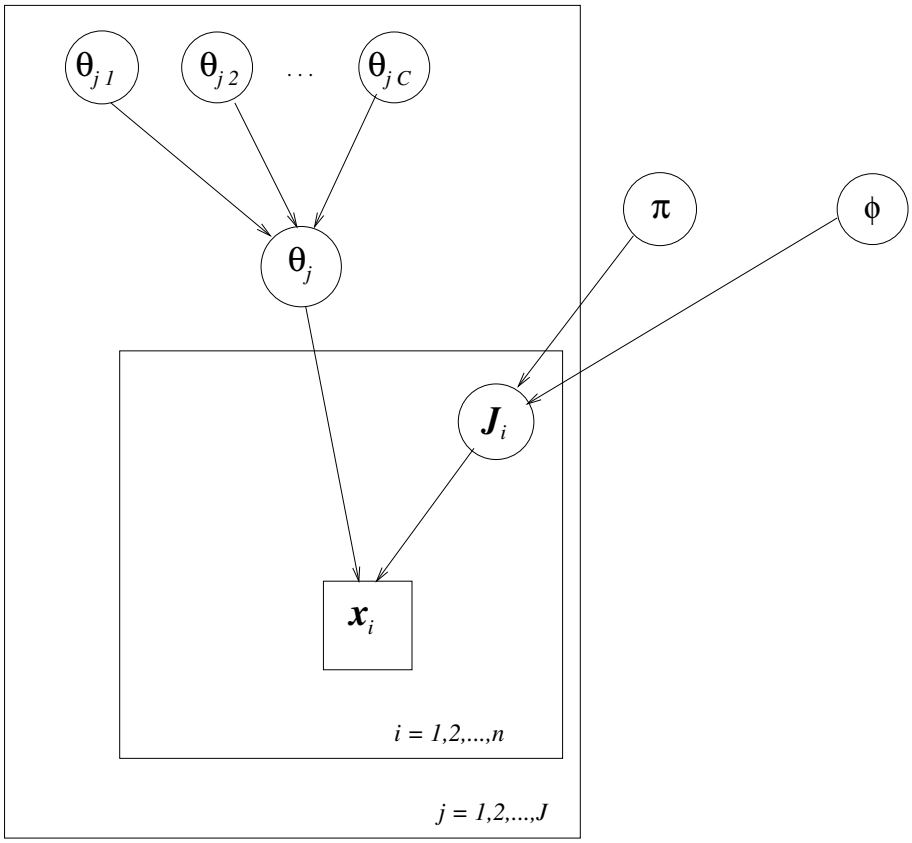

Figure 1: Directed acyclic graph for our model.

They consider a further hierarchy in the parameters by letting $\beta \sim Ga(g, h)$ with g and h fixed (Stephens 1997, also follows this approach), to construct a quasi-non-informative yet proper prior. We follow the same approach here. We fix the following parameters to $\mu_0 = \frac{a+b}{2}$, $g = 0.2$, $\alpha = 2$ $\lambda_0 = \frac{1}{(b-a)^2}$ and $h = \frac{100g}{\alpha(b-a)^2}$, where $a = \min x_{iv}$ and $b = \max x_{iv}$. Richardson and Green (1997) and Stephens (1997) point out that these values convey the belief that the "λ_j's are similar, without being informative about their absolute size". To update μ_j, λ_j and β, we simulate from their full conditionals (Gibbs kernel). That is, a $N(\mu_{n_j}, \lambda_{n_j})$, where $\lambda_{n_j} = \lambda_0 + n_j\lambda_j$ and $\mu_{n_j} = \lambda_{n_j}^{-1}(\lambda_0\mu_0 + n_j\lambda_{n_j}\sum_{J_i=j} x_{iv})$ for μ_j, a $Ga(\alpha + \frac{1}{2}n_j, \beta + \frac{1}{2}\sum_{J_i=j}(x_{iv} - \mu_j)^2)$ for λ_j and a $Ga(g + J\alpha, h + \sum_{j=1}^{J}\lambda_j)$ for β.

3.2. Normal homocedastic

We consider a homocedastic Normal variable, for which $x_{iv} \mid J_i = j, \boldsymbol{\mu}, \boldsymbol{\lambda} \sim N(\mu_j, \lambda)$, with the same precision λ across all groups. It is likely that in clustering problems we would prefer this variable, instead of a heterocedastic one (see Richardson and Green 1997) since it will tend to split the range of the data in more or less uniform intervals. Again, we take $\mu_j \sim N(\mu_0, \lambda_0)$ *a priori*, and $f(\lambda) \propto \lambda^{-1}$ as a prior for λ. To update μ_j we simulate from $N(\mu_{n_j}, \lambda_{n_j})$, where $\lambda_{n_j} = \lambda_0 + n_j\lambda$ and λ from a $Ga(\frac{1}{2}n, \frac{1}{2}\sum_{j=1}^{J}\sum_{J_i=j}(x_{iv} - \mu_j)^2)$ (the full conditionals), which are always proper although the prior for λ is not. As above, we take $\mu_0 = \frac{a+b}{2}$ (the data range mid point) and $\lambda_0 = \frac{1}{(b-a)^2}$, a large precision.

3.3. Multinomial

Regarding Multinomial attributes (eg. categorical), we assume that $x_{iv} = 1, 2, \ldots, L$ (that is, we translate labels into a numeric scale). The conjugate prior is a Dirichlet; however, in this case, there is not a unique standard reference prior, with $Di(1/L, \ldots, 1/L)$ being a common non-informative prior used. From simulated studies, where only one Multinomial variable is considered, we have seen that a $Di(1/2nL, \ldots, 1/2nL)$ leads to a mixture where only L components are effectively used and each component in fact takes only one level. We thus consider the latter as our default (sample size dependent) prior and simulate the vector of Multinomial probabilities in group j from its full conditional (Gibbs kernel), which is a $Di(1/2nL+n_{j1}, 1/2nL+n_{j2}, \ldots, 1/2nL+n_{jL})$, where $n_{jl} = |\{i : x_{iv} = l, J_i = j\}|$ (the current Multinomial counts for group j).

3.4. Poisson

A basic procedure to deal with some types of ordinal variables is to use a Poisson model. We assume that $x_{iv} = 0, 1, \ldots$. Given $x_{iv} \sim Po(\lambda_j)$ the standard reference prior is $f(\lambda_j) \propto \lambda_j^{-1/2}$, which is not proper. Following the hierarchical priors used for the Normal case, we take $\lambda_j \sim Ga(\alpha, \beta)$, fix $\alpha = 1$ and take $f(\beta) \propto \beta^{-1}$ (as a reference prior for β). We then simulate from the full conditional of λ_j which is a $Ga(\alpha + \sum_{J_i=j} x_{iv}, \beta + n_j)$ and we also simulate β from its full conditional (Gibbs kernels), that is a $Ga(J\alpha, \sum_{j=1}^{J} \lambda_j)$.

3.5. Prediction

In the context of MCMC, finding predictive distributions for missing (unobserved) attributes is straightforward. Simply, a missing attribute x_{iv} is taken as an unknown parameter and included in the Markov Chain simulation. From Section 2 we see that the full conditional of x_{iv} is $f(x_{iv} \mid \boldsymbol{\theta}_{J_i v})$, that is, sampling from the model used for attribute v. A "predictive" sample is then obtained for x_{iv}, which can be used either to approximate its predictive distribution or some point estimate like its predictive expectation.

4. Posterior analysis

As in any complex Bayesian analysis, an important problem is analyzing posterior information. **BClass** produces approximations for the posterior grouping probabilities $\boldsymbol{P} = (p_{ij}) = P(J_i = j \mid \boldsymbol{X})$, a $n \times J$ matrix, and posterior expectations for $\boldsymbol{\pi}$ and $\boldsymbol{\theta}$, $E[\pi_j \mid \boldsymbol{X}]$ and $E[\boldsymbol{\theta}_{jv} \mid \boldsymbol{X}]$. As explained earlier one can regard the grouping probabilities as homogeneous attributes. For small n, it is possible to calculate a distance matrix (with simple Euclidean distance) and plot a dendrogram. However, even for moderate n (= 500), the dendrograms are difficult to read since most branches overlap. An alternative and simpler method is to plot the points $a_1 p_{i1} + a_2 p_{i2} + \cdots + a_J p_{iJ}$ where the a_j's are equally spaced along the unit circle. Thus the grouping probabilities for each entry i $(p_{i1}, p_{i2}, \ldots, p_{iJ})$ are mapped within the unit circle and similar grouping probabilities are plotted as nearby points (the contrary is not necessarily true though, and thus some care is needed in the interpretation of the resulting plots). These plots, hereafter referred to as "archipelago" plots, are used in the example to visually recognize clusters. We worked around the problem of cluster definition by plotting the grouping probabilities, say 10 times, with the order of the groups randomized. When

these plots are linked, clusters can be reliably defined as points located nearby in all the plots—false clusters always split apart when permuting the order of the groups.

With respect to the dimensionality variables, we only keep track of $k^{(l)} = \sum_{j=1}^{J} \phi_j^{(l)}$, the number of "alive" groups at pass l. Using the sample $k^{(l)}$ we approximate probabilities for the number of groups used. This is also done in the example.

5. BClass implementation in Lisp-Stat

BClass is fully implemented in Lisp-Stat for several reasons: (1) Lisp programming is remarkably straightforward; (2) it allows to focus on the problem at hand without much concern about the technical innards of the language; (3) most functions are vectorized; and (4) the Lisp-Stat's dynamic graphical-linking capabilities facilitate tremendously the exploratory analysis of both the input data and final results.

BClass was implemented following an object-oriented strategy. Every supported statistical model (i.e. Poisson, multinomial, normal homocedastic and normal heterocedastic) consists of a prototype that can be used to instantiate as many objects as necessary. Therefore, before running **BClass**, a preliminary analysis is required so as to decide which distribution type should be used to model each attribute.

The system includes a comprehensive tutorial plus a demo script to illustrate the readers how to load their own data into **BClass**, fine tune internal parameters (i.e. the number of groups J in the mixture, the number of iterations for the MCMC algorithm, etc.), how to start the classification process, plus how to save or reload the **BClass** output in order to postpone or continue a particular analysis. Two approaches (plot-based and query-based) are combined to define, interpret, and evaluate the quality of the resulting clusters, which are formed by observations sharing similar grouping probabilities. The system is open source code, it consists of 16 script files, the tutorial and the demo. **BClass** is available free of charge from the journal's web page and from the sites `http://www.cimat.mx/~jac/software` and `http://www.ccg.unam.mx/amedrano/BClass`.

6. Case study

In order to understand the instructive example here presented several, biological concepts are necessary. First, DNA is a linear sequence of four types of concatenated building blocks called nucleotides or bases: adenine (A), thymine (T), guanine (G) and cytosine (C). Structurally, DNA is a macromolecule with two complementary strands arranged in a double helix, each strand being read in opposite directions (forward or reverse) by the cellular machinery. Second, a gene is a DNA fragment that encodes the necessary information to produce proteins, which in turn are responsible for carrying out most of the cellular processes indispensable for sustaining life. Genes may be physically positioned in any of the two DNA strands and their length is the sum of As, Ts, Cs, and Gs in their sequence. Third, a bacterial chromosome is a circular self-replicating DNA sequence that contains in a linear array all or most of the genes. Fourth, transcriptional regulation refers to the molecular mechanism responsible for strategically expressing the genes required by the cell in order to survive environmental changes, reproduce, metabolize nutrients, etc. Fifth, the protein product of genes can be classified in several types, here we will deal only with four of them: enzyme (to accelerate biochemical

reactions), regulator (to control whether or not the protein product of other genes will be manufactured), transport (to take nutrients, toxins, etc in and out of the cell), and leader (specific sequences at the beginning of some genes that may function in targeting reactions or regulation). Sixth, gene function in this example refers to metabolic processes in which the genes participate (e.g. amino acid biosynthesis, energy production, cell division, etc.). These definitions, albeit sufficient, are by no means complete.

RegulonDB (Salgado *et al.* 2004) has information on transcriptional regulation for more than 2300 genes. We selected a set of 435 genes for which all attributes are completely described, with the exception of a few missing values (see below). We used the following attributes for the analysis: The DNA strand (forward or reverse), gene length in base pairs (bp), position within the chromosome (in minutes), gene type (enzyme, leader, regulator, transporter and miscellaneous), gene function (20 functional classes) and regulation mode (positive, negative or dual). These attributes, in turn, are modeled as Multinomial with 2 levels, Normal homocedastic, Normal homocedastic, Multinomial with 5 levels, Multinomial with 20 levels and Multinomial with 3 levels, respectively. There are 3 genes that have an unclassified or unknown function, and 11 genes with an unknown mode of regulation. Following the technique explained in Section 3.5 we predicted the values for these missing data. We considered a mixture of $(J =)30$ components.

Given the complexity of the multidimensional model entertained and the fact that Lisp-Stat is an interpreter, the convergence of the MCMC chain turned out to be slow. We took quite a long burn-in of 100,000 sweeps followed by a run of 50,000 sweeps. The π_j's were sorted and sampling was carried out every 5 sweeps. With the current **BClass** implementation in Lisp-Stat, these calculations took nearly 9 hours in a Intel Xeon processor (2.4 GHz) running Linux Red Hat 7.3. In this analysis we concentrated on calculating the grouping probabilities, that is, on obtaining the matrix $\boldsymbol{P} = (p_{ij})$. The archipelago plot for $\boldsymbol{P}$ (see Section 4) is shown in figure 2.

As explained above, using the sample $k^{(l)}$ we approximate probabilities for the number of groups used, see figure 3. We see that the most likely number of components is 29, and thus we have some confidence that we are using an appropriate number of groups.

So far, we have analyzed more than 15 clusters arising from the archipelago plots. For illustration, however, we present a brief analysis of the cluster shown in figure 2. This cluster has 13 genes. We see that all genes are regulated positively (their expression needs to be activated), all but two have a miscellaneous type, all are on the reverse strand, 12 genes participate in central intermediary metabolism (function 3) and the function of one gene is unknown. For this latter gene, *phnQ*, there is a predictive probability of more than 0.98 that its product is involved in central intermediary metabolism (function 3), as the rest of the genes in the cluster. Concerning their position within the chromosome the genes form two clusters: 12 genes form one cluster between minutes 92.93 and 93.11 and one gene, known as *gcvH*, is in minute 65.68. Regarding the size of the genes, they are in the range 309 to 1137 bp (small to average size), with the smallest gene being *gcvH*. In bacteria, a substantial fraction of genes is co-transcribed (expressed at the same time) defining the so-called "operons". All but *gcvH* belong to an operon containing 15 genes. This is the largest known operon in *E. coli* as most operons contain 2 or 3 genes. As far as we know, there is no reported evidence that these genes are functionally linked. Further theoretical and experimental research should be directed at studying the biological meaning of the association between the former gene and the latter operon. A preliminary transcriptome analysis of 4 growth conditions (i.e. heat shock,

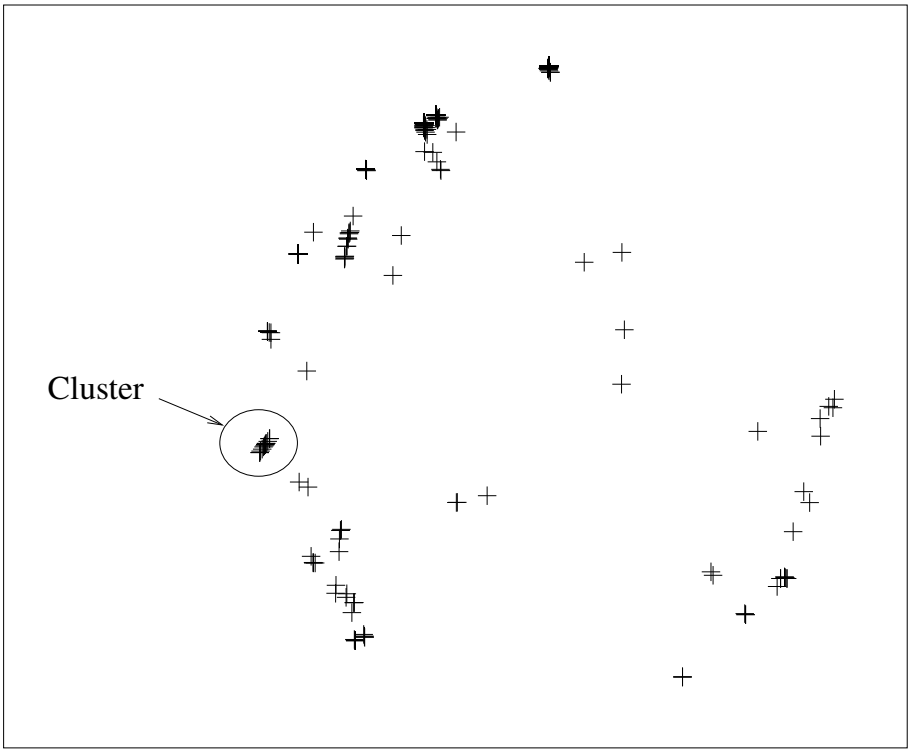

Figure 2: Archipelago plot for the 435 genes analyzed from the **RegulonDB** *E. coli* gene database.

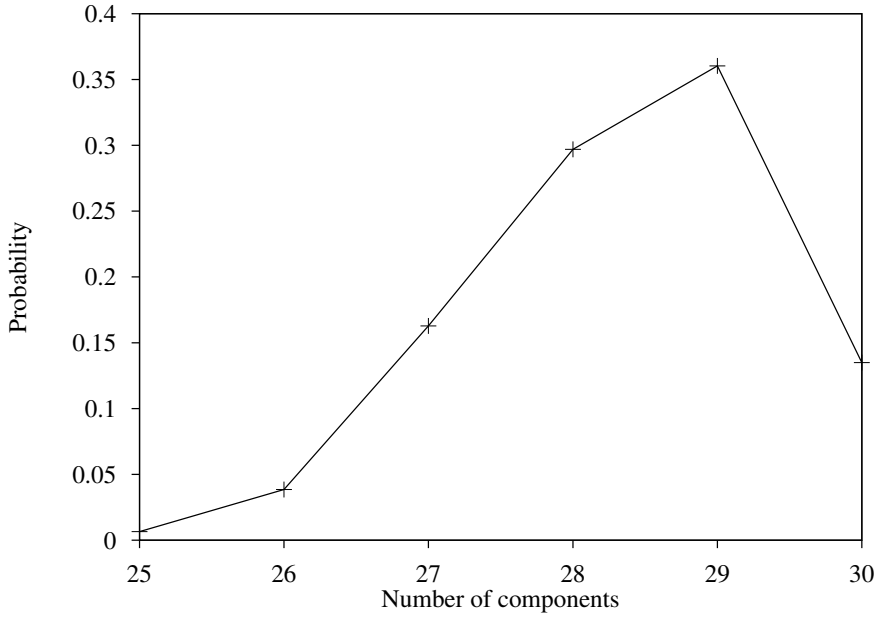

Figure 3: Posterior distribution for the number of "alive" components.

osmotic shock, minimal media, and IPTG) indicates that none of these genes change their expression significantly, so potential coexpression under other non-essayed growth conditions cannot be completely ruled out.

Besides the gene *phnQ* mentioned above, genes *yhdG* and *dsdX* have a predictive probability of nearly 1 that their product is involved in transport (function 14). Regarding the missing values for the type of regulation, gene *treB* has a probability of more than 0.99 of having a dual regulation type. The rest of the missing values analyzed do not have high predictive probabilities. It might be interesting to confirm these predictions with further experimental research.

For the sake of comparison, **AutoClass** was also applied to **RegulonDB**, for exactly the same genes and variables as above. Since **AutoClass** relies on MAP (point) estimates it only reports the most likely group each entry belongs to. **AutoClass** found 7 groups after 10,000 iterations (a minimum of 50 is recommended), however, none of these groups could be identified as one single cluster in the archipelago plot produced by **BClass**. Moreover, the groups produced by **AutoClass** were split into several well compacted more interpretable clusters by **BClass**. Indeed, it is not surprising that we can extract more significant information out of **BClass** since we do have access to the whole vector of grouping probabilities for each entry and not just the most likely group. More comprehensive evaluations are needed to fully address the issue of comparing the performance and capabilities of each software.

7. Discussion

Here we present a general tool for Bayesian classification of genetic databases using mixture models. The methodology is "open" in the sense that, in principle, any set or combination of quantitative genetic attributes may be analyzed, with few new technicalities to be taken care of.

As explained in Section A.2, without any identifiability constraint, the posterior distribution has $J!$ symmetric components. If a MCMC sampler is run without such constraints, samples will be drawn from any of these components. The output of such sampler should be analyzed with much care, avoiding mixing samples from different components. By post-processing the MCMC output, Stephens (1997) proposes two methods to "relabel" a sample from a mixture model. Stephens' second method could be applied here since it is independent of the type of component distributions used. However, this method post-processes the probabilities of the full conditionals for each J_i in every MCMC iteration. This involves keeping s matrices of size $n \times J$, where s is the MCMC sample size; a formidable task even for moderate n. The ordering constraint in the π_j's followed in this paper does avoid some of the label switching problems, provided that the groups do not have similar π_j's. The label switching problem has just recently been addressed, and thus further research efforts are required to find a simpler on-line solution.

The number of components used in the mixture is controlled by the indicator dimensionality variables ϕ_j's, while the entropy prior on the mixing probabilities enable the method to obtain parsimonious parametrizations. We use standard Metropolis-Hastings, which leads to reasonable simple algorithms, in contrast with the complexity of methods proposed elsewhere (Philips and Smith 1996; Richardson and Green 1997; Stephens 1997). Moreover, our approach is independent of the families of component distributions used, making unnecessary

any fine tuning of details for each new distribution introduced.

As discussed above, the rationale behind **BClass** offers several advantages, particularly in the case of genetic databases, compared to other clustering techniques. These ideas should be useful and robust for applications in diverse areas of research such as satellite imaging, social sciences, medicine etc. besides biology.

As far as we know, mixture models have been applied to biology at the level of morphological traits (na and noz 1998) but not to mine databases in molecular biology. Bayesian statistics has been certainly used in Bioinformatics in learning strategies to identify common motifs in related sequences (Neuwald *et al.* 1995), in sequence alignment (Zhu *et al.* 1998), phylogenetics (Lewis and Swofford 2001), as well as in transcriptome analysis (Baldi and Long 2001). Given the explosion and growth of biological databases (see the January issues of the journal Nucleic Acids Research where biological databases are presented), it is reasonable to foresee the increasing importance of mixture models as a tool for clustering analysis and biological knowledge discovery. A well known example of the contribution of clustering techniques to molecular biology is the determination of 3 general codon usage classes in *E. coli* (Médigue *et al.* 1991). These major codon usage groups have been observed in other bacteria as well, which fueled the development of improved computational methods for gene prediction (Borodovsky *et al.* 1995). More recently, the emergence of post-genomic experimental tools has generated an outburst of large data sets of expression profiles for all genes within complete genomes. The virtues of clustering analysis in functional genomics have been clearly illustrated by Eisen *et al.* (1998). Currently, clustering techniques are routinely used in transcriptome analysis to discover genes that might contribute to disease, identify potential drug targets, and sets of corregulated genes that will play an essential role in the eventual characterization of complete genomic regulatory networks (see D'Haeseleer *et al.* 2000). Even though expression values are homogeneous, the methodology here presented permits an integrated analysis through the inclusion of other additional gene attributes. For instance, **BClass** is able to analyze simultaneously transcriptome data, functional categories, regulation modes, position within the chromosome and/or other biological attributes that could strategically improve the search for sets of co-regulated genes. Future work with **BClass** shall illustrate the extent to which heterogeneous classification impinges upon integrative genomic analyses.

The application of **BClass** to a data subset of **RegulonDB** is presented here as a preliminary example. However, it was apparent that the current list-based implementation of **BClass** in Lisp-Stat, albeit functional, is not adequate to analyze large data sets because the code is interpreted. Given that the time required for the execution of the MCMC algorithm is a function of the number of parameters and the number of observations, we have re-implemented the whole system using an array-based approach and a commercial version of common Lisp (Allegro), in order to compile the source code and obtain a fast binary stand-alone application. Here we concentrated on efficiency issues to minimize time-consuming bottlenecks in the code. However, although the overall **BClass** speed was significantly increased, it is not fast enough to handle thousands of entries in a few minutes, often requiring several hours or even days. Given this situation, we are currently working on a parallel version of **BClass** that will be able to run much faster.

Acknowledgements

We thank an anonymous referee for valuable criticisms to the manuscript. AM-S acknowledges a Ph.D fellowship from CONACyT. This work has been supported by grant number 0028 from CONACyT to JC-V. We appreciate computer technical support by V. del Moral and C. Bonavides.

References

Anderson N, Matheson A, Steiner S (2000). "Proteomics: Applications in Basic and Applied Biology." *Current Opinion Biotechnology*, **11**, 408–412.

Baldi P, Long A (2001). "A Bayesian Framework for the Analysis of Microarray Expression Data: Regularized t-Test and Statistical Inferences of Gene Changes." *Bioinformatics*, **17**, 509–519.

Bernardo J, Smith A (1994). *Bayesian Theory*. John Wiley & Sons, Chichester.

Besag J, Green P, Higdon D, Mengersen K (1995). "Bayesian Computation and Stochastic Systems." *Statistical Science*, **10**(1), 3–66.

Borodovsky M, McIninch J, Koonin E, Rudd K, Médigue C, Danchin A (1995). "Detection of New Genes in a Bacterial Genome using Markov Models for three Gene Classes." *Nucleic Acids Research*, **23**(17), 3554–3562.

Brown P, Botstein D (1999). "Exploring the New World of the Genome with DNA Microarrays." *Nature Genetics*, **21**, 33–37.

Cheeseman P, Stutz J (1996). "Bayesian Classification (**AutoClass**): Theory and Results." In PS U Fayyad G Piatetsky-Shapiro, R Uthurusamy (eds.), "Advances in Knowledge Discovery and Data Mining," pp. 153–180. AAAI Press/MIT Press.

Danchin A, Sekowska A (2000). "Expression Profiling in Reference Bacteria: Dreams and Reality." *Genome Biology*, **1**(4), 1024.

D'Haeseleer P, Liang S, Somogyi R (2000). "Genetic Network Inference: From Co-expression Clustering to Reverse Engineering." *Bioinformatics*, **16**, 707–726.

Dutt M, Lee K (2000). "Proteomic Analysis." *Current Opinion Biotechnology*, **11**, 176–179.

Eisen M, Spellman P, Brown P, Botstein D (1998). "Cluster Analysis and Display of Genome-wide Expression Patterns." *Proceedings of the National Academy of Sciences, USA*, **95**(25), 14863–14868.

Everitt B (1993). *Cluster Analysis*. Arnold, London.

Flaischmann R, *et al* (1995). "Whole Genome Random Sequencing and Assembly of *Haemophilus Influenzae*." *Science*, **269**, 496–512.

Fraley C, Raftery A (1998). "How Many Clusters? Which Clustering Method? Answer via Model-based Cluster Analysis." *The Computer Journal*, **41**(8), 579–587.

Gamerman D (1997). *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*. Chapman & Hall, London.

Gordon A (1981). *Classification*. Chapman & Hall, London.

Green P (1995). "Reversible Jump Markov Chain Monte Carlo Computation and Bayesian Model Determination." *Biometrika*, **82**, 711–732.

Hand Y, Keming Y (2001). "Idiot's Bayes—Not so Stupid after all?" *International Statistical Review*, **69**, 385–398.

Kaufman L, Rousseau P (1990). *Finding Groups in Data*. John Wiley & Sons, New York.

Lewis P, Swofford D (2001). "Back to the Future: Bayesian Inference Arrives in Phylogenetics." *Trends in Ecology & Evolution*, **16**(11), 600–601.

McLachlan G, Basford K (1988). *Mixture Models: Inference and Applications to Clustering*. Marcel Dekker, New York.

Médigue C, Rouxel T, Vigier P, Henaut A, Danchin A (1991). "Evidence for Horizontal Gene Transfer in *Escherichia Coli* Speciation." *Journal of Molecular Biology*, **222**(4), 851–856.

na JA, noz MM (1998). "Mixture Analysis in Biology: Scope and Limits." *Journal of Theoretical Biology*, **191**, 341–344.

Neuwald A, Liu J, Lawrence C (1995). "Gibbs Motif Sampling: Detection of Bacterial Outer Membrane Protein Repeats." *Protein Science*, **4**(8), 1618–1632.

Philips D, Smith A (1996). "Bayesian Model Comparison via Jump Diffusions." In SR W R Gilks, D Spiegelhalter (eds.), "Practical Markov Chain Monte Carlo," pp. 441–446. Chapman & Hall, London.

Richardson S, Green P (1997). "On Bayesian Analysis of Mixtures with an Unknown Number of Components." *Journal of the Royal Statistal Society, Series B*, **59**(4), 731–792.

Roeder K, Wasserman L (1997). "Practical Bayesian Density Estimation Using Mixtures of Normals." *Journal of the American Statistal Association*, **92**(439), 894–902.

Salgado H, Gama-Castro S, Martínez-Antonio A, Díaz-Peredo E, Sánchez-Solano F, Peralta-Gil M, García-Alonso D, Jiménez-Jacinto V, Santos-Zavaleta A, Bonavides-Martínez C, Collado-Vides J (2004). "**RegulonDB** (version 4.0): Transcriptional Regulation, Operon Organization and Growth Conditions in *Escherichia Coli* K-12." *Nucleic Acids Research*, **32**, D303–D306.

Stephens M (1997). *Bayesian Methods for Mixtures of Normal Distributions*. Ph.D. thesis, Magdalen College, Oxford.

Titterington D, Smith A, Makov U (1985). *Statistical Analysis of Finite Mixture Distributions*. John Wiley & Sons, New York.

Zhu J, Liu J, Lawrence C (1998). "Bayesian Adaptive Sequence Alignment Algorithms." *Bioinformatics*, **14**(1), 25–39.

A. Technical considerations

A.1. Identifiability and the entropy prior

The problem of identifiability has been discussed extensively in the literature of mixture models (see, for example, Titterington *et al.* 1985). The problem may be described in the following way. Given two sets of parameters $\boldsymbol{h_1}$ and $\boldsymbol{h_2}$ for our mixture model $f(\boldsymbol{X} \mid \boldsymbol{h})$, when is it the case that $f(\boldsymbol{X} \mid \boldsymbol{h_1}) = f(\boldsymbol{X} \mid \boldsymbol{h_2}) \Leftrightarrow \boldsymbol{h_1} = \boldsymbol{h_2}$? (in such circumstances $\boldsymbol{h_1}$ uniquely describes the model and there are no alternative parametrizations). Obviously, if we have $\boldsymbol{h}$ and permute the indices j in $\boldsymbol{h}$, say $\sigma(\boldsymbol{h})$, we have $f(\boldsymbol{X} \mid \boldsymbol{h}) = f(\boldsymbol{X} \mid \sigma(\boldsymbol{h}))$. To avoid this, we only consider labelings in which $\pi_1 \leq \pi_2 \leq \ldots \leq \pi_J$. Now, it is well known, for example, that a mixture of Normals is identifiable, that is $f(\boldsymbol{X} \mid \boldsymbol{h_1}) = f(\boldsymbol{X} \mid \boldsymbol{h_2}) \Leftrightarrow \boldsymbol{h_1} = \boldsymbol{h_2}$ (besides, indeed, label permutations). On the contrary, it is easy to see that in general a mixture of Multinomials is not identifiable (see Titterington *et al.* 1985, p. 35) in which case we have a set H such that $f(\boldsymbol{X} \mid \boldsymbol{h_1}) = f(\boldsymbol{X} \mid \boldsymbol{h_2})$ for $\boldsymbol{h_1}, \quad \boldsymbol{h_2} \in H$. When analyzing heterogeneous databases we will be dealing with Multinomial component distributions and, in general, we do not know which combinations of component distributions or even which families of distributions will be used in the analysis of specific databases. Moreover, even if we were dealing with an identifiable mixture we may still have the problem of *sub-identifiability*. For example, a mixture with five components, $J = 5$, but only four effective groups, $\phi_5 = 0$, could just as well be described with $\phi_5 = 1$, 5 groups. This illustrates two nested models that describe the data equally well.

Therefore, non-identifiability or sub-identifiability (or both) results in a likelihood $f(\boldsymbol{X} \mid \boldsymbol{h})$ with large flat areas. Thus, choosing among alternative parametrizations, could and should be done with an appropriate prior. Suppose then that $\boldsymbol{h_1}$ and $\boldsymbol{h_2}$ are two alternative parametrizations, how would we choose *a priori* among them? Indeed, we would like parsimonious parametrizations, using the least number of groups, or more "compact" groups. In mathematical terms we say that we select the parametrization that has the larger information content (less entropy) in its mixing probabilities. Thus if $\sum_{j=1}^{J} \pi_j^1 \log \pi_j^1 > \sum_{j=1}^{J} \pi_j^2 \log \pi_j^2$ we prefer $\boldsymbol{h_1}$ from $\boldsymbol{h_2}$ (using superindices to distinguish the π_j's).

Finally, to express the above in terms of a prior for $\boldsymbol{\pi}$ we propose the *entropy* prior

$$f(\boldsymbol{\pi}) \propto \exp\left\{\sum_{j=1}^{J} \pi_j \log \pi_j\right\}, \tag{1}$$

for $0 < \pi_1 \leq \pi_2 \leq \ldots \leq \pi_J < 1$ and $\sum_{j=1}^{J} \pi_j = 1$. Thus parametrizations with a higher information content in the mixing probabilities π_j's will have higher prior probability density. From the properties of the information score (see Bernardo and Smith 1994, p. 79) note that the kernel of $f(\boldsymbol{\pi})$ is bounded by 1, therefore $f(\boldsymbol{\pi})$ is well defined.

The natural conjugate prior for $\boldsymbol{\pi}$ is a Dirichlet, the prior used for $\boldsymbol{\pi}$ in virtually all Bayesian mixture model analysis. The entropy prior in (1) is not conjugate but still has a convenient form regarding the MCMC calculations that follow (We may think of a more general form for (1), say $f(\boldsymbol{\pi}) \propto \exp\{\sum_{j=1}^{J}(\alpha_j - 1)\log \pi_j + \gamma_j \pi_j \log \pi_j\}$. This distribution has as a special case the Dirichlet and would be a conjugate prior for $\boldsymbol{\pi}$. However, we decided to use (1) since it represents clearly the information we are trying to convey.).

A.2. MCMC

It is routine in almost any modern Bayesian analysis to use Markov Chain Monte Carlo (MCMC) methods to approximate posterior densities; for a review of the subject see Besag *et al.* (1995). We design our MCMC sampler in the following way. We simulate in turn the parameters $\boldsymbol{\theta}_{jv}$, for $j = 1, 2, \ldots, J$, $v = 1, 2, \ldots, C$ from their full conditionals. The allocation variables J_i's are simulated using a Metropolis step and we simulate $\boldsymbol{\pi}$ and $\boldsymbol{\phi}$ in a single stage using a Metropolis-Hastings step.

It is easy to see that the full conditionals for $\boldsymbol{\theta}_{jv}$ are proportional to $\prod_{J_i=j} f(\boldsymbol{x}_i \mid \boldsymbol{\theta}_{jv}) f(\boldsymbol{\theta}_{jv})$. This means that $\boldsymbol{\theta}_{jv}$ will be drawn as if from a posterior distribution using the likelihood $\prod_{J_i=j} f(\boldsymbol{x}_i \mid \boldsymbol{\theta}_{jv})$ and prior $f(\boldsymbol{\theta}_{jv})$. We will concentrate on conjugate forms for $f(\boldsymbol{\theta}_{jv})$ and, in general, $\boldsymbol{\theta}_{jv}$ will be easy to simulate. There is, however, the difficulty that improper priors may not be used (see Roeder and Wasserman 1997). The definition of $f(\boldsymbol{\theta}_{jv})$ for specific type of variables is left for Section 3.

With respect to the allocation variables $\boldsymbol{J}$, we make a proposal J_i' for J_i with $P(J_i' = j) \propto \phi_j$ (that is, uniformly from the alive components). This represents a Metropolis (symmetrical) proposal and it is not difficult to see that its acceptance probability is

$$\min\left(1, \frac{f(\boldsymbol{x}_i \mid \boldsymbol{\theta}_{J_i'})\pi_{J_i'}}{f(\boldsymbol{x}_i \mid \boldsymbol{\theta}_{J_i})\pi_{J_i}}\right).$$

In our experience, both with simulated and real data, we have seen that this Metropolis step has better mixing than simulating directly from the full conditional of J_i, with the added benefit that only two evaluations of the likelihood are involved.

We construct proposals for $\boldsymbol{\pi}$ and $\boldsymbol{\phi}$ to be accepted or rejected through a Metropolis-Hastings acceptance probability. Let $n_j = |\{i : J_i = j\}|$. We use a Dirichlet $Di(n_1+1, n_2+1, \ldots, n_J+1)$ to simulate a proposal $\boldsymbol{\pi}' = (\pi_1', \pi_2', \ldots, \pi_J')$ for $\boldsymbol{\pi}$. At a first step we ignore the ordering imposed on the π_j's. Due to alternative relabelings, the posterior distribution will have $J!$ symmetric components. Given a simulated value $\boldsymbol{h}^*$ from the posterior, a relabeling of $\boldsymbol{h}^*$ may be considered to be a simulated value from the posterior. We therefore run the MCMC sampler and after a burn-in, when samples may be viewed as drawn from the posterior, we relabel the components to have the correct ordering in the π_j's. After the burn-in we relabel the samples every 5 or 10 passes and use only those samples both to ensure the correct ordering in the π_j's and avoid correlation. Thus in what follows we take the prior for $\boldsymbol{\pi}$ as in (1) ignoring the ordering constraint.

We simulate a proposal $\boldsymbol{\phi}' = (\phi_1', \phi_2', \ldots, \phi_J')$ for $\boldsymbol{\phi}$, independently of $\boldsymbol{\pi}$, using $P(\phi_j' = 1 \mid n_j > 0) = 1$ and $P(\phi_j' = 1 \mid n_j = 0) = \beta$, for some suitable proposal probability β. Note that we are only considering the "birth" or "death" of a group $\phi_j' = 1, 0$ when the group is empty, $n_j = 0$, since a proposal $\phi_j' = 0$ given $n_j > 0$ has zero acceptance probability.

It is proved below that the acceptance ratio for this proposal is

$$A = \exp\left\{u \log\frac{\alpha}{1-\alpha} + w \log\frac{\beta}{1-\beta} + \sum_{j=1}^{J} \pi_j' \log \pi_j' - \pi_j \log \pi_j\right\} \tag{2}$$

where $u = \sum_{j=1}^{J} \phi_j' - \phi_j$ and $w = \nu_0 - \nu_0'$, where $\nu_0 = |\{j : n_j = 0, \phi_j = 0\}|$ and $\nu_0' = |\{j : n_j = 0, \phi_j' = 0\}|$. $\boldsymbol{\pi}$' and $\boldsymbol{\phi}$' are then accepted with probability $\min(1, A)$. We see that it

is not difficult to simulate the proposals and that the acceptance ratio has a rather compact form and is quite simple to calculate.

A.3. Proof of (2)

As explained in Section 2 we use a Dirichlet $Di(n_1+1, n_2+1, \ldots, n_J+1)$ to simulate a proposal $\boldsymbol{\pi}'$ for $\boldsymbol{\pi}$ and we simulate a proposal $\boldsymbol{\phi}'$ for $\boldsymbol{\phi}$, independently of $\boldsymbol{\pi}$, using $P(\phi'_j = 1 \mid n_j > 0) = 1$ and $P(\phi'_j = 1 \mid n_j = 0) = \beta$. For the transition kernel we have that $K\{(\boldsymbol{\pi}, \boldsymbol{\phi}), (\boldsymbol{\pi}', \boldsymbol{\phi}')\} = k_1(\boldsymbol{\pi}')k_2(\boldsymbol{\phi}')$ and $k_1(\boldsymbol{\pi}') \propto \exp\left(\sum_{j=1}^J n_j \log \pi'_j\right)$ and $k_2(\boldsymbol{\phi}') = \exp\left(\nu'_1 \log\beta + \nu'_0 \log\beta\right)$, where $\nu'_h = |\{j : n_j = 0, \phi'_j = h\}|, \quad h = 0, 1$. Noting that $f(\boldsymbol{\pi}, \boldsymbol{\phi} \mid \boldsymbol{X}, \boldsymbol{J}, \boldsymbol{\theta}) \propto f(\boldsymbol{J} \mid \boldsymbol{\pi}, \boldsymbol{\phi}) f(\boldsymbol{\pi}) f(\boldsymbol{\phi})$ and using that $f(\boldsymbol{J} \mid \boldsymbol{\pi}, \boldsymbol{\phi}) = \exp\left(\sum_{j=1}^J n_j \log \pi_j\right)$ for $\phi_j = 1$ such that $n_j > 0$ and zero otherwise, we see that the likelihood is canceled out with $k_1(\boldsymbol{\pi})$ and what is left is the prior ratio and the ratio for $k_2(\boldsymbol{\phi})$. Therefore we obtain

$$
\begin{aligned}
A \quad = \quad & \exp\left(\sum_{j=1}^J \pi'_j \log \pi'_j - \pi_j \log \pi_j\right) \\
& \exp\left(\sum_{j=1}^J (\phi'_j - \phi_j) \log\alpha + (\phi_j - \phi'_j) \log(1-\alpha)\right) \\
& \exp\left((\nu_1 - \nu'_1) \log\beta + (\nu_0 - \nu'_0) \log(1-\beta)\right),
\end{aligned}
$$

with the equivalent definition for ν_1 and ν_0. Letting $n_0 = |\{j : n_j = 0\}|$ we have that $\nu_1 = n_0 - \nu_0$ and it is easy to see that $\nu_0 - \nu'_0 = \nu'_1 - \nu_1 = v$; (2) follows immediately.

Affiliation:

Arturo Medrano-Soto
Program of Computational Genomics
Centro de Ciencias Genómicas (UNAM)
Ave. Universidad s/n, Col. Chamilpa, A.P. 565-A
62100 Cuernavaca, Morelos, Mexico.
E-mail: amedrano@ccg.unam.mx

J. Andrés Christen
Program of Probability and Statistics
Centro de Investigación en Matemáticas, A.C.
A.P. 402,
36000 Guanajuato, Gto., Mexico.
E-mail: jac@cimat.mx

Julio Collado-Vides
Program of Computational Genomics
Centro de Ciencias Genómicas (UNAM)
E-mail: collado@ccg.unam.mx

Journal of Statistical Software
February 2005, Volume 13, Issue 2.
http://www.jstatsoft.org/

Submitted: 2004-04-29
Accepted: 2004-12-20