Some Bayesian biclustering methods: Modeling and inference

Abhishek Chakraborty¹ | Stephen B. Vardeman²

¹Department of Mathematics, Statistics, and Computer Science, Lawrence University, Appleton, Wisconsin
²Department of Statistics and Department of Industrial and Manufacturing Systems Engineering, Iowa State University, Ames, Iowa, USA

Correspondence
Abhishek Chakraborty, Department of Mathematics, Statistics, and Computer Science, Lawrence University, 711 E. Boldt Way – SPC 24, Appleton, WI 54911, USA.
Email: abhishek.chakraborty@lawrence.edu

Abstract

Standard one-way clustering methods form homogeneous groups in a set of objects. Biclustering (or, two-way clustering) methods simultaneously cluster rows and columns of a rectangular data array in such a way that responses are homogeneous for all row-cluster by column-cluster cells. We propose a Bayes methodology for biclustering and corresponding MCMC algorithms. Our method not only identifies homogeneous biclusters, but also provides posterior probabilities that particular instances or features are clustered together. We further extend our proposal to address the biclustering problem under the commonly occurring situation of incomplete datasets. In addition to identifying homogeneous sets of rows and sets of columns, as in the complete data scenario, our approach also generates plausible predictions for missing/unobserved entries in the rectangular data array. Performances of our methodology are illustrated through simulation studies and applications to real datasets.

KEYWORDS
Bayesian analysis, biclustering, MCMC, missing entries, Rand index

1 INTRODUCTION

One typical goal in unsupervised learning is to group either rows (instances) or columns (features) of a rectangular data matrix into homogeneous subsets. In this paper, we will refer to procedures for this problem as one-way clustering procedures (as is done in Reference [1]). Reference [2] provides an overview of some common one-way clustering methods.

In some situations, a rectangular data matrix is two-way, or transposable, meaning that rows and columns are considered to be of equal scientific importance, stimulating the desire to discover structure in both dimensions at the same time. One-way clustering methods implicitly direct the analysis to a particular characteristic (row or column attribute) of the system under study. Biclustering methods perform clustering in two dimensions of the data matrix simultaneously. Discovering simultaneously homogeneous row clusters and column clusters can sometimes answer critical scientific questions besides serving as a tool for exploratory data analysis.

Biclustering methods find applications in a wide array of domains. Cheng and Church [3] were the first to apply biclustering to gene expression data. Since then, most researchers have used gene expression datasets to validate the effectiveness of their biclustering algorithms. In gene expression studies, it is desirable to simultaneously cluster genes and samples/conditions potentially leading
to important biological breakthroughs. Biclustering has been applied in other fields as well. In the field of collaborative filtering, it is desirable to simultaneously identify subgroups of customers with similar preferences or behaviors towards groups of products with the idea of performing target marketing [4]. In information retrieval and text mining, it is desirable to simultaneously identify subgroups of documents with similar properties relative to subgroups of attributes, such as words or images [5]. In agricultural yield studies, it is desirable to simultaneously group plant varieties and growing environments to inform decisions about the yield performance of different varieties of crops at different locations thereby ensuring optimal use of agricultural resources. In developing recommender systems, it is desirable to simultaneously group “raters” and “movies” with the interest of making proper recommendations. Biclustering has also been used to perform dimensionality reduction in databases with tables with thousands of records (rows) and hundreds of fields (columns).

A bicluster is a submatrix of a data matrix consisting of all entries simultaneously in a particular set of rows and a particular set of columns. One searches for biclusters that have similar rows and similar columns. The concept of “similarity” used depends upon the dataset and the scientific question. Some authors also use the term “co-cluster” for a bicluster.

Figure 1 illustrates the arrangement of a typical two-way, or rectangular, dataset. Each element \( a_{ij} \) in the \( n \times m \) matrix depicted in the figure corresponds to a real-valued entry representing a numerical response (like gene expression level) associated with row (“gene”) \( i \) and column (“condition”) \( j \).

The existing approaches to biclustering can be classified in accordance with the type of biclusters identified, the patterns of biclusters discovered, and the methods used to perform the bicluster search [6]. Figure 2 presents the different structures of biclusters that can be identified by existing biclustering algorithms. Comprehensive review of past biclustering algorithms can be found in References [6, 7]. Our present interest is in possibility (c) of the figure, “checkerboard” biclustering.

The rest of this paper is organized as follows. Section 2 details how we formulate the biclustering problem. In Section 3, we propose our Bayesian approach to biclustering. Section 4 contains details about proposed MCMC algorithms. In Section 5, we discuss the identification of a representative bicluster pattern. Section 6 contains the results of simulation studies in the complete data context. We evaluate the performance of our algorithm on a few simulated datasets and present the results in this section. Section 7 presents the results of the application of the algorithm to a real-world lung cancer gene expression data set. We extend our proposed methodology to the incomplete data setting in Section 8 followed by discussions in Section 9.

## 2 | PROBLEM FORMULATION

We will be working with an \( I \times J \) data matrix \( Y \) where an element \( y_{ij} \) will be a real value. For example, in the case of gene expression matrices, \( y_{ij} \) may represent the expression level of gene \( i \) under condition \( j \). For matrices which record movie ratings from customers, \( y_{ij} \) may denote the rating assigned to movie \( j \) by customer \( i \).

The objective of our biclustering methodology is to simultaneously identify the partitions of rows and partitions of columns of the data matrix such that the data entries are similar within each submatrix defined by an element of the row partition and element of the column partition. That is to say, when the rows and columns of the original data matrix are reordered according to their groupings, a checkerboard pattern emerges. The elements within the matrix blocks defined by the row and column groups tend to be relatively homogeneous.

A probabilistic model that can represent such a structure is formulated as follows. Suppose that the latent checkerboard structure is defined by at most \( R \) row groups and \( C \) column groups, that is, we assume that the \( I \) instances can be clustered into at most \( R \) unknown and non-overlapping groups of rows and the \( J \) features can be clustered into at most \( C \) unknown and non-overlapping groups of columns. Hence, we are looking to identify at most \( RC \) biclusters in the data matrix. Let the \( I \)-dimensional vector \( r \) with entries \( r(i) \) in \( \{1,2,...,R\} \) specify the cluster assignment of all rows and the \( J \)-dimensional vector \( c \) with entries \( c(j) \) in \( \{1,2,...,C\} \) specify the cluster assignment of all columns. We adopt the notation \( \mu_{rc} \) for \( RC \) (real) means and assume that \( y_{ij} = \mu_{r(i)c(j)} + \epsilon_{ij} \), where the \( \epsilon_{ij} \) are independent and identically distributed (i.i.d) \( \text{N}(0,\sigma^2) \) variables for some fixed \( \sigma^2 > 0 \). So, we model the entries of the data matrix as

\[
y_{ij}^{\text{ind}} \sim \text{N}(\mu_{r(i)c(j)},\sigma^2),
\]  

(1)
This biclustering model given by form (1) is the same as the model assumed in References [1, 8]. The checkerboard model assumed here is exhaustive in the sense that each data entry belongs to one and only one bicluster. Although each row and each column of the data matrix may have elements in more than one bicluster, the biclusters themselves are non-overlapping. Reference [9] assumed this structure in analyzing cancer data. The Double Conjugated Clustering (DCC) approach introduced in Reference [10] could also find biclusters following this structure. This is in contrast to other biclustering methods (that also assume the existence of several biclusters in the data matrix) that identify potentially overlapping biclusters which are not exhaustive, these are typically estimated using SVD (singular value decomposition)-like methods [11–14] or methods to find hot-spots [15].

3 | METHODOLOGY

Estimating the checkerboard model parameters involves finding the row and column clusters simultaneously and estimating the mean value associated with each bicluster. Thus, the objective is to do inference for the \( \mu_{r(i),c(j)} \), and for the indicators \( 1 (r_i = r_i') \) and \( 1 (c_j = c_j') \). \( 1 (r_i = r_i') \) is a binary variable that indicates whether the \( i \) th row and the \( i \) th row in the data matrix have the same row-cluster membership (1) or not (0). An analogous definition holds for \( 1 (c_j = c_j') \).

We propose a Bayesian approach to solve this biclustering problem. First, we consider the situation where none of the entries in the data matrix are unobserved or missing, that is, we have a complete data array. We extend our algorithm to the incomplete data matrix setting in a later section.

For Bayesian modeling purposes, we consider independent normal priors on the bicluster-specific means, given by

\[
\mu_{rc} \sim N (\mu, \rho^2),
\]

and finite Dirichlet process priors on the row-cluster and column-cluster assignment vectors \( r \) and \( c \) respectively.

We implement the Chinese Restaurant Process (CRP) representation of the Dirichlet process [16], employed in Reference [17]. The CRP is a discrete-time stochastic process analogous to seating customers at tables in a Chinese restaurant. It is strongly connected to the Pólya urn scheme [18], and gives a way to specify a distribution.
over partitions of \( I \) rows (and correspondingly \( J \) columns). Hence, it can be used as a prior distribution on the space of latent variables \( r(i) \) and \( c(j) \) which represent the cluster assignments for the \( i \)th row and \( j \)th column respectively. In the formulation of the Dirichlet process priors detailed below, we choose \( \alpha > 0 \) and \( \beta > 0 \) as prior sample sizes (in terms of the Polya urn scheme) or total prior masses for \( r \) and \( c \) respectively.

Let

\[
n_r(r) = |i : r(i) = r| \quad i = 1, 2, \ldots, I,
\]

\( n_r(r) \) denotes the number of rows out of \( I \) that have class membership \( r \), where \( r = 1, 2, \ldots, R \). We consider a prior distribution for \( r \) with parameter \( \alpha \) given by

\[
g(r) = \left( \prod_{i=1}^{I} \frac{1}{\alpha + (I - i)} \right) \left( \prod_{r \in I, n_r(r) > 0} \left( \frac{\alpha}{R} \right) \left( \frac{\alpha}{R} + 1 \right) \right) \cdots \left( \frac{\alpha}{R} + (n_r(r) - 1) \right).
\]

(3)

Form (3) is symmetric and hence, invariant to permutations of \( r \). Thus we have,

\[
g(r(i) = r_1, \ldots, r(I)) = \frac{\alpha}{R} + \frac{\alpha}{R} + \frac{1}{\alpha + (I - 1)} + \frac{1}{\alpha + (I - 2)} + \cdots + \frac{1}{\alpha + 1}.
\]

(4)

Analogously for column clusters, we define \( n_c(c) = |j : c(j) = c| \cdot j = 1, 2, \ldots, J \), and consider

\[
h(c) = \left( \prod_{j=1}^{J} \frac{1}{\beta + (J - j)} \right) \left( \prod_{c \in C, n_c(c) > 0} \left( \frac{\beta}{C} \right) \left( \frac{\beta}{C} + 1 \right) \right) \cdots \left( \frac{\beta}{C} + (n_c(c) - 1) \right),
\]

(5)

and thus have

\[
h(c(j) = c_1, \ldots, c(J)) = \frac{\beta}{C} + \frac{\beta}{C} + \frac{1}{\beta + (J - 1)} + \frac{1}{\beta + (J - 2)} + \cdots + \frac{1}{\beta + 1}.
\]

Under the model assumptions laid out above, for real values \( \mu \), \( \sigma^2 \), and \( \rho^2 \), the joint distribution for the \( y_{ij} \), the \( \mu_{rc}, \) \( r, \) and \( c \) has density proportional to

\[
g(r)c(h) \prod_{r=1}^{R} \prod_{c=1}^{C} f \left( y_{ij} | \mu_{rc}, \sigma^2 \right) \times f \left( \mu_{rc} | \mu, \rho^2 \right).
\]

(7)

where \( f(\cdot | \mu, \sigma^2) \) denotes the pdf of a normal distribution with mean \( \mu \) and variance \( \sigma^2 \). In the usual definition of a Dirichlet process, the strength, or concentration, parameter \( \alpha \) (and analogously \( \beta \)) serves as a kind of inverse-variance. It determines the extent of repetition of values of the output distribution, which in this case, are the row (and analogously column) class memberships. The larger the value of \( \alpha \), the smaller is the repetition, and vice-versa. In this formulation of the biclustering problem, \( \alpha \) and \( \beta \) control how big the random numbers of non-empty row and column clusters tend to be, small values leading to posterior distributions where \( r \) and \( c \) tend to have few distinct entries. Adjusting the values of \( \alpha \) and \( \beta \) allows us to consider a spectrum of numbers of non-empty row and column clusters, without pre-specifying fixed numbers of row and column clusters.

An alternate version of our Bayes model is formulated by considering an \( R \)-dimensional weight vector \( \pi \) for the rows, and similarly, a \( C \)-dimensional weight vector \( \phi \) for the columns of the matrix. Note that the elements of these two weight vectors must be positive and they must sum to one. The vectors \( \pi = (\pi_1, \ldots, \pi_R) \) and \( \phi = (\phi_1, \ldots, \phi_C) \) are given symmetric Dirichlet prior distributions with concentration parameters \( \alpha/R \) and \( \beta/C \) respectively, that is

\[
p(\pi_1, \ldots, \pi_R | \alpha) \sim \text{Dirichlet}(\alpha/R, \ldots, \alpha/R), \quad p(\phi_1, \ldots, \phi_C | \beta) \sim \text{Dirichlet}(\beta/C, \ldots, \beta/C)
\]

(8)

(9)

In the construction given in forms (8) and (9), the cluster assignment vectors \( r \) and \( c \) function as latent entities. But instead of considering finite Dirichlet process priors on \( r \) and \( c \), we consider symmetric Dirichlet priors for the weight vectors \( \pi \) and \( \phi \). The joint density for the \( y_{ij} \), the \( \mu_{rc}, \) \( \pi, \) and \( \phi \) is proportional to

\[
\left[ \left( \prod_{i=1}^{I} \pi_{r(i)} \right) \left( \prod_{j=1}^{J} \phi_{c(j)} \right) \right] \left[ \prod_{i=1}^{I} \prod_{j=1}^{J} f \left( y_{ij} | \mu_{r(i)c(j)}, \sigma^2 \right) \right] \times \left[ \prod_{r=1}^{R} \prod_{c=1}^{C} f \left( \mu_{rc} | \mu, \rho^2 \right) \right] \times \left[ \frac{\Gamma(\alpha)}{\Gamma(\alpha/R)^R} \right] \left[ \frac{\Gamma(\beta)}{\Gamma(\beta/C)^C} \right] \prod_{r=1}^{R} \prod_{c=1}^{C} \phi_{rc}^{\alpha/\beta - 1}.
\]

(10)

Integrating out \( \pi \) and \( \phi \) from form (10) results in the joint density given in display (7) (Appendix A).
We construct two MCMC samplers (based on the two joint densities in displays (7) and (10)) for the inference of the cluster means $\mu_r$, and the vectors $r$ and $c$. Relative frequencies with which $r(i) = r(i')$ are available which give approximate posterior probabilities that instances $i$ and $i'$ belong to the same row-cluster (the same holds true for features and column-clusters). The two MCMC samplers are detailed in the next section.

## 4 MCMC Samplers

We consider a data matrix $Y$ where none of the $y_{ij}$'s are missing. The MCMC algorithms we employ use Gibbs updates for the parameters. For the Gibbs sampling procedures, we set the value of $\mu$, the prior mean for the bicluster means $\mu_{rc}$; $\rho^2$, the prior variance for the bicluster means; and $\sigma^2$, the data variance. Note that $\mu$ and $\rho^2$ are common for every bicluster. We generally consider the prior mean $\mu$ to be 0 and the prior variance $\rho^2$ to be 100 (large compared to the data variance $\sigma^2$) to make the priors weakly informative [19]. The data variance $\sigma^2$ acts as a tuning parameter. We choose $R$ and $C$ to be fairly large (upper bounds on the numbers of row and column clusters one really expects to have). In general, the values of $R$ and $C$ will vary based on the dimensions of the data matrix to be analyzed and the corresponding subject matter question of interest. We also set the values of $\alpha$ ($>0$) and $\beta$ ($>0$), the total prior masses for the Dirichlet process priors. The samplers are run for $T$ iterations. The first version of the Gibbs sampler corresponding to the joint density in display (7) is presented below.

1. Set $\sigma^2$, $\mu$, $\rho^2$, and the number of iterations $T$.
2. Choose appropriate values of $R$, $C$, $\alpha$, and $\beta$.
3. Initialize $r$, $c$ and $\mu_{rc}$ for $r = 1, 2, \ldots, R$; $c = 1, 2, \ldots, C$. The initial cluster assignments for the rows and columns are separately generated, at random, from discrete uniform distributions on $\{1, 2, \ldots, R\}$ and $\{1, 2, \ldots, C\}$ respectively. Another approach to initialization could be using the $k$-means clustering method separately for the rows and columns. Once the initial cluster assignments for the rows and columns have been generated, the initial value of $\mu_{rc}$ corresponding to bicluster $(r, c)$ is obtained by a random draw from a normal distribution as mentioned in step 4(iii) of the sampler.
4. Repeat $T$ times.

   (i) Conditioned on $c$ and $\mu_{rc}$, for each $i = 1, 2, \ldots, J$, update $r(i)$ one at a time by sampling from the discrete distribution on $\{1, 2, \ldots, R\}$ with probabilities proportional to

   \[ \left( \frac{\alpha}{R} + |\{i': i' \neq i \text{ with } r(i') = r\}| \right) \times \prod_{c=1}^{C} \prod_{j \text{ s.t. } c(j) = c} f(y_{ij}|\mu_{rc}, \sigma^2), \]

   (ii) Conditioned on $r$ and $\mu_{rc}$, for each $j = 1, 2, \ldots, J$, update $c(j)$ one at a time by sampling from the discrete distribution on $\{1, 2, \ldots, C\}$ with probabilities proportional to

   \[ \left( \frac{\beta}{C} + |\{j': j' \neq j \text{ with } c(j') = c\}| \right) \times \prod_{r=1}^{R} \prod_{i \text{ s.t. } r(i) = r} f(y_{ij}|\mu_{rc}, \sigma^2), \]

   (iii) Conditioned on $r$ and $c$, each $\mu_{rc}$ is updated by sampling from the posterior distribution generated from the normal prior and the sample of size $n_{rc}$ of entries $y_{ij}$ for those $(i, j)$ currently with $(r(i), c(j)) = (r, c)$.

   \[ \mu_{rc}| y_{ij}, r, c \sim N \left( \frac{\mu}{\rho^2} + \frac{n_{rc} \cdot \tilde{y}_{rc}}{\sigma^2}, \left( \frac{1}{\rho^2} + \frac{n_{rc}}{\sigma^2} \right)^{-1} \right), \]

   \[ \times \left( \frac{1}{\rho^2} + \frac{n_{rc}}{\sigma^2} \right)^{-1}, \]

   where

   \[ \tilde{y}_{rc} = \left( \sum_{(i, j) : r(i), c(j) = (r, c)} y_{ij} \right) / n_{rc}, \]

   and

   \[ n_{rc} = | \{y_{ij} : (i, j) \text{ s.t. } r(i), c(j) = (r, c)\} |; \]

   for $r = 1, 2, \ldots, R$, $c = 1, 2, \ldots, C$.

A second version of the Gibbs sampler based on the joint density in display (10) is next.

1. Set $\sigma^2$, $\mu$, $\rho^2$, and the number of iterations $T$.
2. Choose appropriate values of $R$, $C$, $\alpha$, and $\beta$.
3. Initialize $r$, $c$ and $\mu_{rc}$ for $r = 1, 2, \ldots, R$; $c = 1, 2, \ldots, C$.

   The initialization of $r$, $c$ and $\mu_{rc}$ can be done as in Step (3) of the first version described above. Initialize $\pi = (\pi_1, \ldots, \pi_R) \sim \text{Dirichlet}(\alpha/R, \ldots, \alpha/R)$ and $\phi = (\phi_1, \ldots, \phi_C) \sim \text{Dirichlet}(\beta/C, \ldots, \beta/C)$.

4. Repeat $T$ times.

   (i) Conditioned on $r$ and $c$, update $\pi = (\pi_1, \ldots, \pi_R)$ and $\phi = (\phi_1, \ldots, \phi_C)$ by sampling from $\text{Dirichlet} \left( \frac{\alpha}{R} + n_1(r), \ldots, \frac{\alpha}{R} + n_R(r) \right)$ and $\text{Dirichlet} \left( \frac{\beta}{C} + n_1(c), \ldots, \frac{\beta}{C} + n_C(c) \right)$ respectively. $n_r(r)$,
for \( r = 1, \ldots, R \) and \( n_c(c) \), for \( c = 1, \ldots, C \) have been defined in Section 3.

(ii) Conditioned on \( c, \pi, \) and \( \mu_{rc} \), the \( r(i) \)'s are sampled independently, each from \( \{1,2,\ldots,R\} \), with probabilities proportional to

\[
\pi_r \prod_{c=1}^{C} \left( \prod_{j \in 1, c(j) = c} f(y_{ij} | \mu_{rc}, \sigma^2) \right).
\]

(iii) Conditioned on \( r, \phi, \) and \( \mu_{rc} \), the \( c(j) \)'s are sampled independently, each from \( \{1,2,\ldots,C\} \), with probabilities proportional to

\[
\phi_r \prod_{r=1}^{R} \left( \prod_{i \in 1, r(i) = r} f(y_{ij} | \mu_{rc}, \sigma^2) \right).
\]

(iv) Conditioned on \( r \) and \( e \), each \( \mu_{rc} \) is updated by sampling from a normal posterior as mentioned in step 4(iii) of the first version of the sampler.

Note that in the second version of the sampler, the \( r(i) \)'s and \( c(j) \)'s can be sampled all at once unlike the one-at-a-time update in the first version.

\section{Identifying a Representative Bicluster Pattern}

The complexity of the biclustering problem is seen in the number of possible solutions to the corresponding partitioning problem. The total number of partitions of an \( n \)-element set is given by the Bell number, \( B_n \). For a dataset that is considered small by current standards, say a \( 15 \times 10 \) data matrix, the number of possible partitions is given by \( B_{15} \times B_{10} \), which is in the order of magnitude of \( 10^{14} \). With current computational resources, it is infeasible to go through such a huge space of all possible combinations of row and column partitions to identify a set of biclusters typical of the posterior distribution. Therefore, to choose a combination of row and column partitions potentially “representative” of the posterior we choose an MCMC iterate at which an appropriate objective function is optimized.

The algorithm is run for a sufficiently large number of complete Gibbs iterations to explore as much of the partition space as possible. The consideration of the Dirichlet process priors on vectors \( r \) and \( e \) coupled with the choice of \( R \) and \( C \) as possible upper bounds on the numbers of row and column clusters respectively allow the algorithm the flexibility to choose different numbers of row and column clusters at different iterations. Under our biclustering model framework, it is reasonable to choose the total Sum of Squared Errors (SSE), summed over all the biclusters formed, as an objective function. We record the total SSE at each iteration and choose as a representative pattern, the combination of row and column partitions identified at the iterate with the smallest total SSE. The SSE for the \( (r,c) \)th bicluster at each iteration is computed as

\[
SSE_{rc} = \sum_{(i,j) \text{ s.t. } r(i),c(j) = (r,c)} (y_{ij} - \bar{y}_{rc})^2,
\]

where \( \bar{y}_{rc} \) is defined in step 4(iii) of the first MCMC sampler.

The total SSE at each iteration is then computed as

\[
SSE = \sum_{r=1}^{p} \sum_{c=1}^{q} SSE_{rc},
\]

where \( p \) and \( q \) are the numbers of non-empty row and column clusters respectively, determined at that particular Gibbs iteration.

The criterion based on the sum of squared errors represents a measure of similarity between the elements within a bicluster. However, it tends to identify very fine row and column partitions thereby resulting in a large number of biclusters (close to the upper bound \( RC \)). It is also possible that in certain applications the total SSE drops down to 0 where every single cell is a bicluster of itself. A potential improvement in this criterion’s performance can be obtained by including a complexity term which penalizes for the number of biclusters identified. One can thus consider the penalized SSE (pSSE) given by

\[
pSSE = \log_{10}(SSE + 1) + pg,
\]

where \( p \) and \( q \) have been defined previously. This modified metric retains the simplicity of the previous criterion as it can be calculated at every iteration of the MCMC sampler without any additional operations. The combination of row and column partitions identified at the iterate with the smallest \( pSSE \) can then be considered as a representative bicluster pattern.

Another reasonable objective function to identify a representative bicluster pattern is based on the Rand index [20]. The Rand index maps a pair of partitions to a number between 0 and 1, where 1 indicates perfect agreement between the two data clusterings. Observe that the row (and, correspondingly column) partitions obtained from an MCMC sampler represent a distribution over row (column) clusterings. We compute the Rand index between two row (column) clusterings obtained at two different MCMC iterations. A representative row (column) partition might be the one with the highest average Rand index when compared with other row (column) partitions across all iterations. The corresponding set of biclusters obtained by combining the row and column partitions having the respective highest average Rand index is determined as a
representative bicluster pattern. This criterion can also be applied to the element-wise partitions resulting from the MCMC run. The combination of the row and column clusterings corresponding to the MCMC iterate that results in the element-wise partition having the highest average Rand index when compared with other element-wise partitions across all iterations can be considered as a representative bicluster pattern.

The MCMC iterates represent in approximate form a posterior distribution over the row and column partitions. The Rand index metric has been considered to interpret and summarize the clustering results as a decision. The highest average Rand index criterion assigns to each row partition a numerical value which indicates its relative similarity across partitions defined over all iterates. The row partition with the highest average Rand index specifies the row clustering that is most similar with all other row clusterings obtained across all iterations. Thus, it can be interpreted as the single most likely partition of the rows given the data and the model. The Rand index measure essentially considers how each pair of rows is assigned in each clustering. Observe that the clusters in a row partition are defined just as much by those rows which they do not contain as by those which they do contain. If the rows in an individual row-pair are placed together in the same cluster in each of the two clusterings being compared, or if they are assigned to different clusters in both clusterings, this represents a similarity between the clusterings as opposed to the scenario where the rows are clustered together in one clustering and are in different clusters in the other. From this perspective, the criterion represents a measure of how regularly across all iterations row-pairs have been placed together in the same cluster or assigned to different clusters normalized by the total number of row-pairs. The same argument can be extended for column partitions. The most likely row and column partitions are chosen separately, that is, the row and column partitions with the respective highest average Rand index may or may not come from the same iterate, unlike in the total SSE (or, penalized SSE) criterion. A representation of the underlying bicluster pattern in the data as captured by the sampler is then obtained as the cross-product of the most likely row and column partitions.

Continuing with the objective of deciding a representative bicluster pattern, we consider a third criterion based on the joint posterior distribution of the model parameters conditioned on the data entries $y_{ij}$. The joint posterior distribution in this setup is proportional to either form (7) or form (10). It is reasonable to consider the iterate at which the log of the joint posterior distribution is maximized, the combination of the corresponding row and column partitions identifying a representative bicluster pattern. This criterion is equivalent to the maximum a posteriori probability estimate of a bicluster pattern based on the empirical results of the MCMC sampler.

It is possible to develop other optimizing criteria or different versions of the ones mentioned above. However, we work with the proposed objective functions for the simulation studies discussed next.

## 6 | SIMULATION STUDIES

We investigate the performance of our biclustering algorithm with the following simulation studies.

1. Analysis of the performance of the algorithm on simulated datasets.
2. Comparison with independent one-way $k$-means clustering of the rows and columns.
3. Comparison with sparse biclustering approach.
4. Multiplicative biclusters setting.
5. Overlapping multiplicative biclusters setting.

In all our simulation studies, heatmaps showing respective bicluster patterns are arranged in terms of the row and column cluster means. The cluster means increase from bottom to top for row clusters and from left to right for column clusters respectively.

For the following analyses, both versions of the MCMC sampler were implemented. The results were similar in terms of the chains of total SSE values and log posterior values, the respective Rand indices, the representative bicluster patterns identified, and the summaries for distinct combinations of numbers of non-empty row and column clusters found. We report the results from the implementation of the first sampler.

## 6.1 | Analysis of the performance of the algorithm on simulated datasets

For this simulation study, we assess the biclustering performance of the algorithm and examine its computational complexity. We consider the following three data generating situations.

1. $50 \times 60$ data matrix $Y$ generated from 2 row clusters and 3 column clusters with $y_{ij} \sim N(\mu_{rc}, 1^2)$, where $\mu_{rc} \sim \text{Unif}(-3, 3)$, $r = 1, 2$, and $c = 1, 2, 3$. We implement the algorithm with $R = 5, C = 5, \alpha = 10, \beta = 10, \sigma = 1, \mu = 0$, and $\rho = 10$.

2. $100 \times 60$ data matrix $Y$ having 5 row clusters and 3 column clusters with $y_{ij} \sim N(\mu_{rc}, 3^2)$, where $\mu_{rc} \sim \text{Unif}(-3, 3)$, $r = 1, 2, \ldots, 5$, and $c = 1, 2, 3$. We implement the algorithm with $R = 10, C = 5, \alpha = 10, \beta = 10, \sigma = 3, \mu = 0$, and $\rho = 10$. 

For practical purposes, after the MCMC chain has been run for $T$ complete iterations, we record the numbers of non-empty row and column clusters obtained at every iteration, and for each distinct combination of the two, we report the frequency, the minimum, maximum, and mean total SSE, the minimum, maximum, and mean log posterior value, and the minimum penalized SSE value. An illustration of this is presented in a following simulation study. It is left to the researcher to choose an appropriate representative bicluster pattern based on the task at hand.

6.1.1 Biclustering performance

The Rand index is used as a measure for evaluating the biclustering performance of the algorithm. We report the Rand index summaries for each of the three different criteria discussed in Section 5, namely, the minimum total SSE (SSE), the highest average Rand index across all iterations (avgRI), and the maximum logarithm of the joint posterior distribution of model parameters conditioned on the data entries $(lp)$. These three representative clustering patterns, corresponding to the three different criteria, are then compared with the “true” row, column, and element-wise clusterings and the Rand index is calculated between them. The Rand index summaries are reported separately for row, column, and element-wise partitions.

All three criteria discussed above perform well in identifying the “true” latent bicluster pattern within the simulated datasets as can be observed from the Rand indices being close to 1. The bicluster pattern identified using the highest average Rand index criterion (avgRI) reflects better the “true” bicluster pattern in the sense that the representative row, column, and element-wise clusterings identified with this criteria agree with the “true” row, column, and element-wise partition more accurately than the representative clusterings identified using the other two optimizing criteria. Moreover, it is anticipated that the minimum total SSE (SSE) and the maximum log posterior (lp) criteria will identify biclusters where the numbers of non-empty row and column clusters are close to the upper bounds $R$ and $C$ respectively. These two criteria tend to discover finer row and column partitions resulting in a higher number of biclusters with the bicluster sizes being small than the avgRI criterion which tends to discover the more commonly occurring clustering pattern across all MCMC iterations. In such cases, one is recommended to use the criterion based on penalized SSE which adjusts for the number of biclusters identified as mentioned in Section 5.

For practical purposes, after the MCMC chain has been run for $T$ complete iterations, we record the numbers of non-empty row and column clusters obtained at every iteration, and for each distinct combination of the two, we report the frequency, the minimum, maximum, and mean total SSE, the minimum, maximum, and mean log posterior value, and the minimum penalized SSE value. An illustration of this is presented in a following simulation study. It is left to the researcher to choose an appropriate representative bicluster pattern based on the task at hand.

6.1.2 Computational complexity

The runtime of the algorithm scales linearly with the total number of complete iterations $T$ and almost linearly with the size of the data matrix in terms of the total number of entries $I \cdot J$. The computational complexity of the algorithm can be represented as approximately $\mathcal{O}(T \cdot I \cdot J)$. There is negligible difference in the runtimes between the two versions of the MCMC sampler. These results are based on the implementation of the algorithm (both versions) written in R on a Condo Cluster compute node with two 2.6 GHz 8-core Intel E5-2640 v3 Haswell processors.

It is to be noted that the functionality to compute the average Rand index for a row (and correspondingly column) clustering observed at a particular MCMC iterate against all other iterates in the run scales approximately as $\mathcal{O}(T^2)$. For a $20 \times 20$ data matrix and $T = 10,000$, a C++ implementation of the above functionality for one dimension takes approximately 30 minutes on a Condo cluster node. However, this is much faster when compared to the equivalent R function which takes approximately 70 min with the same hardware specifications. The compute time increases with increase in size of the chosen dimension as well. Hence, it takes much longer to compute the average Rand index values for the element-wise partitions. For these reasons, the computational complexity required to identify a representative bicluster pattern using the highest average Rand index criterion is much more than the complexity of the functionalities involving the minimum total SSE and maximum log posterior criteria. In fact, the total SSE and the log posterior values are computed along the MCMC run itself thus requiring no additional computation.

6.1.3 Performance on a simulated dataset

We simulated a $100 \times 200$ data matrix $Y$, where $y_{ij}^{\text{ind}} \sim N(\mu_{rc}, 5^2)$ with $\mu_{rc} \sim \text{Unif}(-3, 3)$. The original data matrix contains 5 row clusters and 5 column clusters, therefore,
The data generating mechanism has been reproduced from one of the simulation studies in [1]. We implemented our biclustering algorithm with \( R = 10, C = 10, \alpha = 10, \) and \( \beta = 10. \) We chose \( \alpha = 5, \mu = 0, \rho = 10, \) and ran the MCMC chain for \( T = 10,000 \) iterations. The heatmaps of the raw data matrix and the bicluster patterns identified according to the minimum total SSE (SSE), the highest average Rand index (avgRI), and the maximum log posterior (lp) criteria are displayed in the Data S1 document. The row and column partitions identified according to the highest average Rand index and the minimum penalized SSE criteria were identical.

It is evident that the minimum total SSE and the maximum log posterior criteria identify a higher number of biclusters within the data matrix compared to the highest average Rand index criterion. However, the bicluster pattern identified according to the latter agrees more accurately with the original bicluster pattern in the data matrix. We also report the summaries of total SSE, log posterior, and penalized SSE values in the MCMC run against each distinct combination of numbers of non-empty row and column clusters obtained. According to the minimum total SSE and maximum log posterior criteria the numbers of non-empty row and column clusters forming the representative bicluster patterns are (7,9) and (10,10) respectively, which is close to the specified values of \( R = 10 \) and \( C = 10. \) These result in finer row and column partitions. On the other hand, the highest average Rand index criterion identifies a bicluster pattern formed from a combination of (5,5) row-column clusters. Note that the data matrix has been generated from 5 row clusters and 5 column clusters.

To check for convergence, we implemented five runs of the algorithm using the above 100 \( \times \) 200 data matrix with different initializations of vectors \( \mathbf{r} \) and \( \mathbf{c} \) respectively. This implies that the bicluster means \( \mu_{rc} \) for \( r = 1,2, \ldots, R; \) \( c = 1,2, \ldots, C \) are also initialized differently. To examine the entire paths of the five MCMC chains and thus, the algorithm itself, the graphs and the quantitative results reported below are based on all 10,000 complete iterations. For each run, the bicluster patterns identified according to the highest average Rand index and the minimum penalized SSE criteria were identical.

The plots of total SSE values against iterates for the five runs and the running mean plot suggest that the total SSE value stabilizes around the 495,000 mark. The initial drop in the first few iterations is caused due to the numbers of non-empty row and column clusters being high, close to the specified upper bounds of \( R \) and \( C \) respectively, which after subsequent iterations adjusts to having reduced numbers of non-empty row and column clusters without showing marked deviations from the minimum value.

We use Geweke’s convergence diagnostic [21] to examine the convergence of each MCMC chain as seen through the total SSE values. The convergence diagnostic by Geweke is based on a test for equality of means of the first and last part of a Markov chain. For the total SSE plots we use the default, that is, the first 10% and the last 50% of the chain. The two means are equal if the samples are drawn from the stationary distribution of the chain, and Geweke’s statistic has an asymptotically standard normal distribution. The test statistic reported is a standard Z-score, which is evaluated as the ratio of the difference between the two sample means to its standard error. The standard error is estimated from the spectral density at zero, and so, considers any autocorrelation. It is assumed that the two parts of the chain are asymptotically independent when calculating the Z-score. Convergence issues arise when the value of the test statistic lies beyond \( \pm \)2. The test statistic values for the five runs of the algorithm based on the total SSE are 0.627, 0.381, 0.679, 0.049, and 0.690.

Another quantitative effort to assess convergence is suggested in Reference [22]. The Gelman-Rubin diagnostic (or, potential scale reduction factor) is applicable in this context since it requires that multiple chains be run from different starting points. The potential scale reduction factor (PSRF) is based on a comparison of within-chain and between-chain variances similar to a classical analysis of variance. Convergence is diagnosed when the chains have forgotten their initial values, and the output from all chains is almost identical. Quantitatively, values of PSRF substantially above 1 indicate lack of convergence (approximate convergence is also diagnosed when the upper limit is close to 1). For the chains of total SSE values, the PSRF is 1.01 with the upper limit equal to 1.01.

Similar to the total SSE plots the log posterior values reach a maximum at the first few iterations and then stabilize around the 490 mark. The test statistic values for Geweke’s convergence diagnostic are 1.400, 0.831, 0.907, 0.691, and -0.318. The PSRF and its upper limit is 1. Both the graphical and quantitative diagnostics show promising signs of convergence. The starting points do not seem to affect the behavior of the algorithm. There appears to be good mixing of the chains for both total SSE and log posterior values, and the running means appear to have reached a steady state.

Furthermore, we compute the Rand index between the row, column, and element-wise clustering at every iteration and the corresponding “true” clustering pattern within the data matrix. We also compute respective Rand indices between clusterings observed at two successive iterates. These Rand index values are plotted on separate histograms for each run of the algorithm. It can be seen that even though the algorithm starts at different points, it eventually identifies the row-column partition that agrees...
TABLE 1  For each run of the algorithm, reported are the Rand index values computed between the “true” row, column and element-wise clustering and the representative row, column, and element-wise clustering having the highest average Rand index across all iterates

| Run | Row | Column | Element-wise |
|-----|-----|--------|--------------|
| 1   | 1   | 0.98   | 0.99         |
| 2   | 1   | 0.98   | 1            |
| 3   | 1   | 0.98   | 0.99         |
| 4   | 1   | 0.98   | 1            |
| 5   | 1   | 0.98   | 0.99         |

well with the latent “true” bicluster pattern. Table 1 reports the Rand indices computed between the “true” row, column and element-wise clustering and the representative row, column, and element-wise clustering having the highest average Rand index across all iterates in a particular run of the algorithm.

Another technique we use to check whether the MCMC chains have attained a stable representation is through the running plots of the numbers of non-empty row and column clusters. The combination of (5,5) and (5,6) non-empty row and column clusters seem to be the two most frequent clustering combinations even though R and C are specified to be much higher. It is to be noted that the “true” number of row and column clusters is 5 each. It is observed that the proposed biclustering algorithm stabilizes after a few iterations and the eventual representative bicluster pattern is a good representation of the “true” bicluster pattern within the data matrix.

In addition to recovering the latent bicluster structure within the data matrix, the methodology involved in our biclustering approach also allows us to compute the empirical frequency, from the MCMC run, with which two rows (or columns) are clustered together. This serves as an estimate of the posterior probability that particular instances (or features) are grouped together, which is especially important when the researcher is interested in the behavior of a few specific instances (or features). This information can be obtained from the row and column cluster frequency plots.

6.2  Comparison with independent one-way k-means clustering

This simulation setting is similar to a simulation study conducted in Reference [1]. To compare the performance of our biclustering approach with independent one-way k-means clustering of the rows and columns, we consider the same data matrix \(Y\) as used in simulation study 6.1.3. We performed independent one-way 5-means clustering on the rows and columns of \(Y\), as well as implemented our biclustering method with \(R = 10, C = 10, a = 10,\) and \(\beta = 10.\) We chose \(\sigma = 5, \mu = 0, \rho = 10\) and ran the algorithm (first version of the sampler) for \(T = 10,000\) iterations.

We consider the biclusters identified according to the highest average Rand index criterion as the representative pattern. The proposed methodology identifies a more accurate bicluster pattern in terms of recovering the latent checkerboard pattern of the means matrix compared to independent one-way 5-means clustering of the rows and columns. Our algorithm identifies homogeneous, non-overlapping, and exhaustive biclusters, and can be implemented with relaxed assumptions in the sense that even without a definite specification of the numbers of row and column clusters it is able to recover the latent bicluster pattern of the original data matrix. For k-means clustering, one needs to specify \(k\) which is generally unavailable. Even if we had known the “true” number of row and column clusters, which is 5 in this case, our algorithm performs better than independent one-way k-means clustering approach.

6.3  Comparison with sparse biclustering

In this simulation study, we compare our approach with sparse biclustering, a widely used biclustering technique developed in Reference [1]. The sparse biclustering technique is comparable to our approach in the sense that it also assumes an underlying checkerboard mean structure. We use the same 100 \(\times\) 200 data matrix \(Y\) as used in simulation studies 6.1.3 and 6.2. We implemented our biclustering algorithm with identical parameter values as in simulation 6.2. The sparse biclustering algorithm was implemented in R using the sparseBC package. The number of row and column clusters were both set to 5. The non-negative regularization parameter was set to 0, implying no regularization.

Both these methods are equally accurate in discovering the latent bicluster pattern defined by the “true” row and column groups. Both correctly partition the rows to their “true” clusters. There are 5 columns, 4 in the fourth column cluster and 1 in the fifth column cluster, which are incorrectly clustered by both these methods. It can be observed that our methodology performs equally well under a more relaxed set of assumptions as compared to the sparse biclustering technique. One can specify plausible upper bounds on the numbers of row and column clusters and still achieve accurate results.
6.4 Multicative biclusters

This simulation study is a replication of a study conducted in [1, 13]. We consider a setup where the data matrix contains multiplicative biclusters. The underlying means matrix is given by a $100 \times 50$ matrix $\mathbf{M} = du_1v_1^T$ with $d = 50$, and

$$\tilde{u}_1 = [10, 9, 8, 7, 6, 5, 4, 3, \text{rep}(2, 17), \text{rep}(0, 75)]^T,$$

$$\tilde{v}_1 = [10, -10, 8, -8, 5, -5, \text{rep}(3, 5), \text{rep}(-3, 5), \text{rep}(0, 34)]^T,$$

where $\text{rep}(a, b)$ denotes a vector of length $b$, all of whose entries are $a$. The means matrix is formed using the normalized vectors $u_1 = \tilde{u}_1/\|\tilde{u}_1\|_2$, and $v_1 = \tilde{v}_1/\|\tilde{v}_1\|_2$. The structure of the means matrix implies that every row and column is generated by multiplying a constant term to every other row and column respectively. The data matrix is generated as $\mathbf{Y} = \mathbf{M} + \epsilon$ where $\epsilon$ is a $100 \times 50$ matrix with elements $\epsilon_{ij}$ that are i.i.d $N(0, 1)$.

We implemented our biclustering algorithm on $\mathbf{Y}$ for $T = 10,000$ iterations with $R = 3, C = 5, \alpha = 1$, and $\beta = 1$. We chose $R = 3$ and $C = 5$ based on the simulation study conducted in Reference [1] in an identical setting, where they consider a spectrum of possible values for the numbers of row and column clusters and select an optimal pair via cross-validation. The data variance was chosen to be $\sigma = 1$, and the values of the prior parameters for the bicluster means were chosen to be $\mu = 0$ and $\rho = 10$. The heatmaps of the data matrix and the bicluster pattern identified by our approach are reported in the Data S1 document. The bicluster patterns identified according to the minimum total SSE and the highest average Rand index criteria were identical, closely resembling the pattern obtained using the minimum penalized SSE criterion.

The data generating mechanism in this simulation study matches with the framework for the sparse Singular Value Decomposition (sparseSVD) technique for biclustering developed in Reference [13]. Therefore, it can be expected that the sparseSVD approach will have the best results in this simulation setup. On the other hand, this is a challenging problem for our proposed biclustering method due to the presence of multiplicative biclusters. Even though our method is designed for a homogeneous and contiguous bicluster setting with an underlying checkerboard mean structure, it successfully identifies the latent bicluster formed from the zero elements of the means matrix $\mathbf{M}$. This might be because the multiplicative biclusters can be approximated as the union of several constant biclusters [1].

From the total SSE plot and histograms of Rand indices computed between two successive iterates, it is seen that the sampler attains stability after a few initial iterations. The column partitions do not change much after the initial iterations. The row partitions stabilize around the halfway point of the chain. The cluster frequency plots for the rows and columns also indicate that the algorithm successfully identifies the bicluster of zero elements in the underlying means matrix as can be observed from the submatrices at the bottom right.

6.5 Overlapping multiplicative biclusters

In this simulation study, we investigate the performance of our proposed biclustering approach on a dataset containing a pair of overlapping multiplicative biclusters [1, 8, 13]. This is an example where our assumption of an underlying checkerboard pattern with non-overlapping biclusters is clearly violated. The underlying means matrix is given by $\mathbf{M} = du_1v_1^T + du_2v_2^T$, where $d = 50$. $u_1, u_2, v_1,$ and $v_2$ are defined as normalized vectors of the form $u = \tilde{u}/\|\tilde{u}\|_2$ and $v = \tilde{v}/\|\tilde{v}\|_2$ where $\tilde{u}_1$ and $\tilde{v}_1$ are as defined in simulation study 6.4, and

$$\tilde{u}_2 = [\text{rep}(0, 13), 10, 9, 8, 7, 6, 5, 4, 3, \text{rep}(2, 17), \text{rep}(0, 62)]^T,$$

$$\tilde{v}_2 = [\text{rep}(0, 9), 10, -9, 8, -7, 6, -5, \text{rep}(4, 5),$$

$$\text{rep}(-3, 5), \text{rep}(0, 25)]^T,$$

where $\text{rep}(a, b)$ denotes a vector of length $b$, all of whose entries are $a$. The $100 \times 50$ data matrix is generated as $\mathbf{Y} = \mathbf{M} + \epsilon$ where $\epsilon$ is a matrix of elements $\epsilon_{ij}$ that are i.i.d $N(0, 1)$.

We implemented our biclustering algorithm for $T = 10,000$ iterations with $R = 3, C = 6, \alpha = 10$, and $\beta = 10$. The choices of $R$ and $C$ were obtained from a study conducted on the same dataset in [1]. The data variance was chosen to be $\sigma = 1$, and $\mu$ and $\rho$ were chosen to be 0 and 10 respectively. Both versions of the MCMC sampler reported similar results, we present the results from the first sampler. The figures related to this analysis are displayed in the Data S1 document. The row partitions identified according to the highest avgRI and minimum penalized SSE criteria were identical, however, there were differences in the respective column partitions.

Our algorithm is not equipped to recover overlapping biclusters and struggles in that respect as can be observed from the resulting bicluster pattern identified. A similar observation was also noted in References [1, 8]. Given the misspecified model, the estimated means matrix is unable to represent the true underlying means structure well. However, convergence does not seem to be an issue even when the model is misspecified with a non-checkerboard structure. One can still approximate posterior probabilities of rows and columns being clustered together from the corresponding frequency plots.
7  |  APPLICATION TO A REAL DATASET

To illustrate the performance of our biclustering approach on a real dataset, we consider the lung cancer gene expression data previously studied in References [1, 8, 13, 23]. Biclustering methods have been commonly used in biological data analysis. The focus has typically been on two-way gene expression datasets resulting from microarray experiments. In these datasets, each gene corresponds to a row and each condition is associated with a column or vice-versa. Each element of the data matrix is a real-valued entry representing the expression level of a gene under a specific condition. The conditions may correspond to different time points, or different experimental or environmental conditions, or may originate from different organs, or even different individuals or samples. The lung cancer gene expression dataset contains measurements for 56 samples (along columns) and gene expression values (along rows) of a subset of 200 genes having the highest variance out of the 12,625 genes measured using the Affymetrix 95av2 GeneChip.

Although a particular type of cancer, such as lung cancer, may present itself as a homogeneous disease, there usually exist multiple distinct subtypes in terms of the gene expression levels. A fundamental objective of applying biclustering on these datasets from cancer research studies is to identify sets of genes that are expressed similarly in certain subtypes, thereby highlighting the association between groups of genes and subtypes [24, 25]. Discovering such biologically relevant patterns is a crucial first step toward developing individualized treatment solutions specific to a patient’s particular cancer subtype.

The dataset analyzed can be found in the R package s4vd. The samples comprise 20 pulmonary carcinoid samples (C), 13 colon cancer metastasis samples (Col), 17 normal lung samples (N), and 6 small cell lung carcinoma samples (SC). The rownames are affymetrix gene ids. Initially, we applied our algorithm with \( R = 10, C = 5, \alpha = 10, \beta = 10, \) and \( \sigma = 1 \) (approximately the standard deviation of the data entries). The parameters \( \mu \) and \( \rho \) were chosen to allow the algorithm to explore as much of the row-column partition space as possible. The prior parameters \( \mu \) and \( \rho \) were specified to be 0 and 10 respectively to reflect our lack of substantial prior knowledge about the bicluster means. These values were intended to capture a small amount of real-world information (weakly informative), thereby allowing the data to primarily guide our inferences. Practitioners can alter the above specifications based on their domain-specific understanding of the data.

The algorithm was run for \( T = 10,000 \) iterations. Both versions of the MCMC sampler were implemented and the results were similar. We report the results for the first sampler. The heatmap of the dataset, the bicluster pattern identified according to the highest average Rand index criterion, and related plots for this analysis are available in the Data S1 document. Figure 3 displays the cluster frequency plot for samples obtained with the above set of parameter values. We want to point out that our biclustering approach is used as an unsupervised learning tool in the sense that we use the cancer subtype information only after the algorithm has been run to interpret the results of our analysis and assess the performance of our algorithm.

We also examined the performance of the algorithm for different \((R,C)\) pairs, \((R = 12, C = 5)\) and \((R = 15, C = 8)\), holding all other parameters fixed. All three heatmaps depicting the bicluster patterns obtained for the three different \((R,C)\) pairs have been arranged in ascending order in terms of the row and column cluster means, bottom row cluster, and left column cluster having smallest means respectively. For each \((R,C)\) specification, the bicluster patterns identified according to the highest average Rand index and the minimum penalized SSE criteria were similar in terms of the Rand index between them.

The heatmaps clearly indicate associations between the distinct blocks of genes and sample groups that emerge from our analysis. There is substantial variation in the expression levels of groups of genes among the different types of samples. There exist sets of genes whose expression values are high for normal samples, whereas the expression values are on the lower side for carcinoid samples and vice versa. There are subsets of genes for which the colon cancer and small cell samples tend to have similar expression values that lie between those for carcinoid and normal samples. For some other groups, the expression values for colon cancer and small cell samples are higher than those for carcinoid and normal samples. Furthermore, the groups of genes correspond to informative grouping of the 56 samples. The four types of samples tend to get clustered into their respective groups, however, we do observe one subject with small cell carcinoma being mostly assigned with the pulmonary carcinoid samples. The bicluster patterns identified reveal important information about the behavior, in terms of expression levels, of groups of genes across groups of samples.

8  |  BAYESIAN BICLUSTERING OF DATASETS WITH MISSING ENTRIES

It is not uncommon that some data entries \( y_{ij} \) in a rectangular data array \( Y \) are missing, or remain unobserved or unreported. Missing values result in datasets for various reasons in different scientific fields. In some experimental settings, the missingness can be attributed to time and/or cost constraints, and in some
other scenarios, the element \( y_{ij} \) representing the relation between row \( i \) and column \( j \) is just not observed. For example, in agricultural yield studies, it is not expected that all the varieties of a crop under study will be tested at every location under investigation. For datasets such as those that record movie ratings from customers, not all data entries \( y_{ij} \), the rating assigned to movie \( j \) by customer \( i \) (or vice versa), are observed. In fact, a very large proportion of data entries are missing. The possibility that there is no numerical response observed for some \((i,j)\) is not uninformative and really demands some kind of attention. Moreover, the biclustering problem becomes more difficult with increase in sparsity of the data matrix.

In the biclustering context, [3] defined a mean squared residue score assuming there are no missing values in the dataset. They replaced the missing values by random numbers during a data preprocessing phase to guarantee this assumption. [26] developed the FFlexible Overlapped biClustering (FLOC) algorithm, based on the bicluster definition used in Reference [3], that dealt with the missing values by introducing an occupancy threshold. Reference [27] proposed handling missing data in microarray datasets by treating them as additional unknown variables and iteratively imputing them in the Gibbs sampling iterations. Most other existing biclustering algorithms require complete data matrices. We consider the missingness mechanism as being informative and attempt to model that.

### 8.1 Methodology

Suppose that every \((i,j)\) has an associated \( y_{ij} \), but that some of them are missing (or unobserved). Instead of treating the missingness as randomly generated, we undertake the assumption that the data entries \( y_{ij} \) which are supposed to be at the lower end of the spectrum of possible values for the variable \( Y \) have a higher probability of remaining unobserved or being missing than the rest of the data entries. In that regard, if a response \( y_{ij} \) is supposed to be small, the probability of it being missing is high, that is, the higher is the corresponding probability of not observing it. This scenario is common in datasets arising from recommender system studies and studies related to agricultural yield of plant varieties. For instance, if the outcome of a plant variety is expected to be low at a particular environmental condition then that variety-condition combination is avoided and the corresponding data entry remains unobserved. We model the above assumption through the following methodology.

We consider the “s-shaped” logistic function given by \( l(z) = \frac{\exp(z)}{1 + \exp(z)} \). Continuing with that, we consider the
two-parameter family of functions
\[ p(z|\lambda_1, \lambda_2) = t \left( \frac{z - \lambda_1}{\lambda_2} \right), \] (13)

Suppose, as mentioned before, that conditioned on the bicluster means \( \mu_{rc} \), the \( y_{ij} \) are independent with means \( \mu_{r(i),c(j)} \) and now that with conditional probability
\[ 1 - p \left( y_{ij}|\lambda_1, \lambda_2 \right), \]

response \( y_{ij} \) is missing (or unobserved). In this setting, the joint density of all \( I \times J \) responses, both observed and unobserved, is
\[
\prod_{r=1}^{R} \prod_{c=1}^{C} \left( \prod_{(i,j) \in T(r,c)} f \left( y_{ij}|\mu_{rc}, \sigma^2 \right) \right) \times \left( \prod_{y_{ij}\text{ observed}} p \left( y_{ij}|\lambda_1, \lambda_2 \right) \right) \left( \prod_{y_{ij}\text{ unobserved}} \left( 1 - p \left( y_{ij}|\lambda_1, \lambda_2 \right) \right) \right) .
\] (14)

Form (14) is an extension of the likelihood term in the complete (non-missing) data setting which was the first term.

Similar to the complete data matrix setting as described in Section 3, we consider independent normal priors on the bicluster-specific means as in form (2), and finite Dirichlet process priors on the row-cluster and column-cluster assignment vectors as in forms (3) and (5) respectively. In addition, to account for the missing data entries in \( Y \), we also consider a normal prior on the model parameter \( \lambda_1 \) and independently, after a log transformation of \( \lambda_2 \) \( \geq 0 \) to \( \log(\lambda_2) \in \mathbb{R} \) to employ a convenient Metropolis step in the MCMC sampler detailed below, a normal prior on \( \log(\lambda_2) \) given by
\[
\lambda_1 \sim N \left( 0, \rho_{\lambda_1}^2 \right),
\] (15)
\[
\log(\lambda_2) \sim N \left( 0, \rho_{\lambda_2}^2 \right),
\] (16)
respectively.

In this paper, we are primarily interested in recovering a representative bicluster pattern, estimating the bicluster means, and generating plausible predictions for the missing entries. Interested practitioners can draw inferences for the parameters \( \lambda_1 \) and \( \lambda_2 \) (or, \( \log(\lambda_2) \)) from posterior samples generated during the MCMC run. In practice, there might be little known about \( \lambda_1 \) and \( \lambda_2 \), thus weakly informative normal priors with variances set to some large value seem to be reasonable and convenient. Researchers with sufficient prior knowledge about the data and related studies can alter these values and incorporate more accurate prior information.

### 8.2 An MCMC sampler

The MCMC sampler constructed for the missing data setting differs from the first version of the sampler for complete data setting introduced in Section 4 in the steps that refer to updates for the parameters \( \lambda_1 \) and \( \lambda_2 \), and updates for the missing \( y_{ij} \). Respective Metropolis steps are used to update \( \lambda_1 \), \( \log(\lambda_2) \), and missing \( y_{ij} \). The row-cluster and column-cluster assignment vectors, and the bicluster-specific means are still updated using Gibbs steps. Similar to the complete data setting, we consider \( R \) and \( C \) to be fairly large, typically upper bounds on the numbers of row and column clusters one really expects to have. We also set the values of \( \alpha (>0) \) and \( \beta (>0) \), the total prior masses for the Dirichlet process priors. The algorithm is run for \( T \) iterations. The MCMC sampler, a Metropolis-within-Gibbs sampler, for the biclustering algorithm addressing missing entries is presented below.

1. Set the model parameters \( \sigma^2, \mu, \rho_1^2, \rho_2^2 \), and the number of iterations \( T \). Choose the standard deviations \( \gamma_{1,1}, \gamma_{1,2}, \) and \( \gamma_{2} \).
2. Choose appropriate values of \( R, C, \alpha, \) and \( \beta \).
3. Initialize unobserved/missing \( y_{ij} \) to complete the data matrix. The starting values for unobserved/missing \( y_{ij} \) can be estimated by the average of observed entries in the data matrix. The researcher can also assign starting values to the missing \( y_{ij} \) based on expertise and experience gained from past scientific experiments.
4. Initialize \( r, c, \lambda_1, \log(\lambda_2), \) and \( \mu_{rc} \) for \( r = 1, 2, \ldots, R; \) \( c = 1, 2, \ldots, C \). The initial cluster assignments for the rows and columns are separately generated, either at random from discrete uniform distributions on \( \{1, 2, \ldots, R\} \) and \( \{1, 2, \ldots, C\} \) respectively, or by using the k-means clustering method. Once the initial cluster assignments are determined and starting values are assigned to the missing \( y_{ij} \), the initial value of \( \mu_{rc} \) corresponding to the bicluster \((r,c)\) can be obtained using step 5(iv) of the sampler below. \( \lambda_1 \) and \( \lambda_2 \) can be initialized as 0 and 1 (hence, \( \log(\lambda_2) = 0 \)) respectively.
5. Repeat \( T \) times.

(i) Conditioned on \( y_{ij} \), update \( \lambda_1 \) and \( \log(\lambda_2) \) using a density proportional to
\[
\left( \prod_{y_{ij}\text{ observed}} p \left( y_{ij}|\lambda_1, \lambda_2 \right) \right) \left( \prod_{y_{ij}\text{ unobserved}} \left( 1 - p \left( y_{ij}|\lambda_1, \lambda_2 \right) \right) \right) \times f \left( \lambda_1|0, \rho_{\lambda_1}^2 \right) f \left( \log(\lambda_2)|0, \rho_{\lambda_2}^2 \right),
\]
\( \lambda_1 \) and \( \log(\lambda_2) \) are updated using two successive univariate Metropolis steps. The proposals are generated from normal distributions centered at the value of the corresponding
current iterate with standard deviations $\gamma_{1}\lambda_{1}$ and $\gamma_{2}\lambda_{2}$ respectively.

(ii) Conditioned on $y_{ij}$, $\mu_{rc}$, and $c$, for each $i = 1, 2, \ldots, I$, the $r(i)$'s are updated one at a time following step 4(i) of the first version of the sampler for non-missing data.

(iii) Conditioned on $y_{ij}$, $\mu_{rc}$, and $r$, for each $j = 1, 2, \ldots, J$, the $c(j)$'s are updated one at a time following step 4(ii) of the first version of the sampler for non-missing data.

(iv) Conditioned on $y_{ij}$, $r$ and $c$, updates for the bicluster means $\mu_{rc}$ are still straight Gibbs draws based on conjugacy, that is, from respective posterior distributions generated from normal priors and samples of corresponding entries $y_{ij}$, identical to step 4(iii) of the first version of the sampler for the complete data setting.

(v) Conditioned on everything else, an unobserved/missing $y_{ij}$ is updated using a density proportional to

$$f(y_{ij}|\mu_{r(i)c(j)}, \sigma^{2})(1 - p(y_{ij}|\lambda_{1}, \lambda_{2}))$$

A missing $y_{ij}$ is updated via a Metropolis step by proposing a value from a normal distribution centered at the value of the current iterate with fixed standard deviation $\gamma_{y}$. Here, $f(\cdot|\mu, \sigma^{2})$ denotes the normal pdf with mean $\mu$ and variance $\sigma^{2}$.

### 8.3 Identifying a representative bicluster pattern and prediction

Analogous to the complete data framework where there were no missing entries, we can choose a representative bicluster pattern to be the one that minimizes the total Sum of Squared Error (SSE) across all $T$ complete MCMC iterations. The expression for the total SSE at each iteration is given in form (12), but it is to be noted that when the data matrix has missing entries, the SSE for each bicluster at an iteration is calculated based only on the observed entries within that bicluster. The total SSE can be adjusted for the number of biclusters identified by adding a penalty term as mentioned in Section 5. A representative bicluster pattern can also be obtained from the combination of row and column clusterings having the respective highest average Rand index across all iterations. Furthermore, when the dataset has missing entries, the joint posterior distribution of the missing $y_{ij}$ and the model parameters conditioned on the observed data entries is proportional to

$$g(r|h(c))$$

$$\prod_{r=1}^{R} \prod_{c=1}^{C} f(y_{ij}|\mu_{rc}, \sigma^{2})$$

$$\times f(\mu_{rc}|\mu, \rho^{2}) \prod_{y_{ij} \text{ observed}} p(y_{ij}|\lambda_{1}, \lambda_{2})$$

where $f(\cdot|\mu, \sigma^{2})$ denotes the pdf of a normal distribution with mean $\mu$ and variance $\sigma^{2}$. We consider the iterate at which the log of the joint posterior distribution in display (17) is maximized, the corresponding row and column partitions identifying a representative bicluster pattern. This is similar to the maximum log posterior criterion described in Section 5 for the complete data setting.

At any iteration of the MCMC sampler, the prediction for an unobserved $y_{ij}$ is given by the mean of the corresponding bicluster (obtained from the posterior distribution) to which that element is assigned to. If we suppose that $\hat{y}_{ij}^{r}$ denotes the predicted value for a missing $y_{ij}$ at an MCMC iterate $t$, then

$$\hat{y}_{ij}^{r} = \mu_{r(i)c(j)}$$

for $t = 1, \ldots, T$.

The average of the predictions across all iterations given by

$$\hat{y}_{ij} = \frac{\sum_{t=1}^{T} \hat{y}_{ij}^{r}}{T}$$

can be considered as a plausible prediction for an unobserved $y_{ij}$ after an MCMC run.

### 8.4 Computational complexity

The computational complexity of the biclustering algorithm in the complete data setting was approximately $O(T \cdot I \cdot J)$, where $T$ is the number of complete MCMC iterations, $I$ and $J$ are the row and column dimensions of the data matrix respectively. The complexity continues to hold in the missing data context as well. The algorithm runtime also increases with increase in the percentage of missing entries within the dataset. It is however difficult to formulate exactly how the varying missing value percentages affect the algorithm runtime. From simulations it is seen that for a fixed number of iterations $T$, the runtime is around 1.5–2 times higher for approximately 20% missingness, about 2.5–3 times higher for approximately 50% missingness, and approximately 3.5–4.5 times higher when the missingness is approximately 90% compared to the runtime for a complete dataset across different matrix dimensions. It is also observed that for a fixed missing value percentage the ratio of the runtimes for different values of $T$ decreases with increase in the matrix dimension.
8.5 Application to a real dataset

We implemented our biclustering methodology on a real-life dataset shared with us by a leading agricultural organization. We extend our sincere gratitude to them for their contribution to this research study. The dataset consists of agricultural yield measurements for 132 different varieties of a commodity of interest at 73 separate locations. The varieties are arranged along the rows and the locations form the columns.

Each cell in the data matrix represents the yield of a particular variety of the commodity at a specific location. As can be seen from the heatmap of the dataset in the Data S1 document, not all cells are filled with a data entry. The percentage of unobserved entries is almost 73%. This dataset reflects well our underlying assumption about the missingness mechanism. Due to time and cost constraints, it is not possible to observe the yield of every variety at every environmental condition. Further, it is assumed that the yield of certain varieties at some locations will be unobserved if those varieties are expected to perform poorly at those specific locations.

With these assumptions in place, we implemented the algorithm on the above dataset with \( R = 15, C = 15, \alpha = 20, \) and \( \beta = 20. \) We chose the data variance parameter \( \sigma^2 \) to be 9. The parameters \( \alpha \) and \( \beta \) were chosen to allow the algorithm to explore as much of the row-column partition space as possible. We ran the MCMC chain for \( T = 10,000 \) iterations with \( \mu = 0, \rho = 10, \rho_{21} = 10, \) and \( \rho_{22} = 10. \) These prior parameters were intended for the respective priors to be weakly informative so that inferences in this analysis are mostly guided by information from the data. Domain-specific expert knowledge can be used to consider other values for these parameters. We initialized the parameters \( \lambda_1 = 0 \) and \( \lambda_2 = 1. \) The standard deviations of the normal densities used as proposals for the Metropolis updates of missing \( y_{ij}, \lambda_1, \) and \( \log(\lambda_2) \) were taken to be \( \gamma_y = 3, \gamma_{\lambda_1} = 2, \) and \( \gamma_{\lambda_2} = 1 \) respectively.

We also implemented a missing-at-random version of the biclustering algorithm with identical initial cluster assignments. The parameters \( \lambda_1 \) and \( \lambda_2 \) are not involved in the missing-at-random version. The rest of the parameter values were kept the same. An unobserved \( y_{ij} \) was updated by drawing randomly from a normal distribution centered at the posterior mean of the bicluster to which it belonged with standard deviation \( \sigma. \) The detailed results of this study are reported in the Data S1 document. The bicluster patterns identified according to the highest average Rand index and the minimum penalized SSE criteria were similar in terms of the Rand index between them.

The analyses with our assumption of informative missingness and the missing-at-random setting yield different bicluster patterns with different respective predictions. The proposed algorithm based on informative missingness is able to group the low-performing and the high-yielding varieties with their corresponding locations into separate biclusters. Because of the high missing value percentage for this data matrix, there are multiple biclusters which are found to be empty.

9 DISCUSSION

The biclustering algorithm proposed and formulated in the previous sections is characterized by identification of homogeneous, non-overlapping, and exhaustive biclusters. Unlike most existing biclustering techniques which are heuristic in nature, our proposed Bayesian approach offers a principled solution to the biclustering problem. The performance of the algorithm in identifying the underlying latent bicluster pattern within a data matrix is demonstrated through simulation studies and applications to real datasets. The results of the algorithm are easily interpretable since it explicitly returns the rows and columns of the data matrix that are clustered together unlike some of the existing biclustering approaches. Besides returning the set of biclusters, due to the Markov chain nature of our algorithm, we can also compute empirical probabilities of two rows, or columns, or elements, being clustered together after an MCMC run. These serve as estimates of posterior probabilities that two rows, or columns, or elements, are grouped together.

The use of the Dirichlet process priors for the row-cluster and column-cluster assignment vectors (or equivalently, the use of the Dirichlet distribution as a prior on weight vectors for rows and columns) enables the algorithm to perform a stochastic search over the space of different combinations of numbers of non-empty row and column clusters. The algorithm only requires to specify plausible upper bounds on numbers of row and column clusters rather than fixed pre-determined numbers of clusters. Across a chain of complete MCMC iterations, the algorithm traverses through distinct combinations of numbers of non-empty row and column clusters capturing a range of behavior of the row and column partitions. The parameters \( \alpha \) and \( \beta \) help to control the variability in the row and column cluster memberships. Smaller values of these parameters result in predominantly fewer numbers of non-empty row and column clusters. As \( \alpha \) and \( \beta \) become large, the finite Dirichlet process prior tends to behave similarly to the uniform prior on the cluster assignment vectors resulting in larger numbers of non-empty row and column clusters. The values of \( \alpha \) and \( \beta \) can be chosen separately and they need not be equal. Practitioners can use their domain-specific knowledge about the data and the particular objective of the biclustering task to choose
these values. Based on the model framework and observations from simulation studies, we recommend choosing large $\alpha$ and $\beta$, such that the ratios $\alpha/R$ and $\beta/C$ are equal to or exceeds 1, to explore biclusters formed from combinations of large numbers of non-empty row and column clusters. Small values of $\alpha$ and $\beta$, or equivalently $\alpha/R$ and $\beta/C$ close to zero, lead to row and column partitions with few non-empty components, which could be relevant if the focus is on investigating a small number of biclusters within the data matrix.

We also propose three separate criteria to identify a representative bicluster pattern. The bicluster pattern chosen by minimizing the total SSE across all iterations tends to have the numbers of non-empty row and column clusters close to the respective upper bounds, while the highest average Rand index criterion tends to select the most frequent bicluster pattern among the MCMC iterates. The total SSE criterion can be adjusted for the number of biclusters obtained and the resulting metric can be used instead. These criteria are intended to assist the researcher in selecting from bicluster patterns of different nature. One can obviously motivate and develop other criteria, or formulate different versions of the ones mentioned in Section 5. The biclustering algorithm developed in the previous sections does not require any preprocessing of the original data matrix and with sufficient computational resources can be implemented on datasets of dimensions higher than those used in the simulations above. A limitation of our proposed biclustering approach is that it is not designed to identify overlapping biclusters. While recovering overlapping biclusters can be beneficial in certain cases [6], models that allow such a structure tend to be too complex and difficult to interpret [1, 8]. However, it remains an interesting direction of future research to modify our proposed approach to elegantly deal with such underlying bicluster patterns.

Another feature of our proposed biclustering approach is that it can be implemented on datasets with missing or unobserved entries. Having missing entries in two-way, rectangular datasets is not uncommon. Through our approach, we have attempted to model the missingness mechanism and performed biclustering even when the matrix is not complete. This is unlike most of the existing biclustering methods which either require the data matrix to be complete, or they impute the missing values randomly, or completely ignore the cells which are missing. The proposed algorithm identifies reasonable biclusters even when a large percentage of entries are missing. We are also able to generate plausible predictions for the missing entries at the end of the MCMC run. All the other properties of the algorithm in the complete data setting are applicable in the incomplete dataset scenario as well. The predicted values from our algorithm can be used for imputation or as initial values of missing entries for different scientific experiments requiring similar information. It is possible to develop other models to tackle the informative missingness apart from the one mentioned here, yet our methodology could serve as a stepping stone towards developing modified biclustering techniques capable of working with incomplete datasets.

**CONFLICT OF INTEREST**
The authors declare no potential conflict of interests.

**DATA AVAILABILITY STATEMENT**
The R codes, datasets, and supplemental material relevant to this article are publicly available at https://github.com/abhicc/Bayesian-Biclustering.git.

**ACKNOWLEDGMENT**
Open access funding provided by the Iowa State University Library.

**ORCID**
Abhishek Chakraborty https://orcid.org/0000-0002-8460-6696

**REFERENCES**
1. K. M. Tan and D. M. Witten, *Sparse biclustering of transposable data*, J. Comput. Graph. Stat. 23 (2014), no. 4, 985–1008.
2. T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: Data mining, inference and prediction*, 2nd ed., Springer, New York, 2009.
3. Y. Cheng and G. M. Church, 2000: Biclustering of expression data. In Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology, 8, 93–103.
4. T. Hofmann and J. Puzicha, 1999: Latent class models for collaborative filtering. In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, 99, 688–693.
5. I. S. Dhillon, 2001: Co-clustering documents and words using bipartite spectral graph partitioning. In Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 269–274.
6. S. C. Madeira and A. L. Oliveira, *Biclustering algorithms for biological data analysis: A survey*, IEEE/ACM Trans. Comput. Biol. Bioinform. 1 (2004), no. 1, 24–45.
7. A. Tanay, R. Sharan, and R. Shamir, *Biclustering algorithms: A survey*, Handbook of Computational Molecular Biology, Chapman and Hall/CRC, New York, 2005.
8. E. C. Chi, G. I. Allen, and R. G. Baraniuk, *Convex biclustering*, Biometrics 73 (2017), no. 1, 10–19.
9. Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein, *Spectral biclustering of microarray data: Co-clustering genes and conditions*, Genome Res. 13 (2003), no. 4, 703–716.
10. S. Busygin, G. Jacobsen, and E. Krämer, 2002: Double conjugated clustering applied to leukemia microarray data. In 2nd
11. S. Bergmann, J. Ihmels, and N. Barkai, *Iterative signature algorithm for the analysis of large-scale gene expression data*, Phys. Rev. E 67 (2003), no. 3, 031902.

12. L. Lazeroni and A. Owen, *Plaid models for gene expression data*, Stat. Sin. 12 (2002), no. 1, 61–86.

13. M. Lee, H. Shen, J. Z. Huang, and J. S. Marron, *Biclustering via sparse singular value decomposition*, Biometrics 66 (2010), no. 4, 1087–1095.

14. M. Sill, S. Kaiser, A. Benner, and A. Kopp-Schneider, *Robust biclustering by sparse singular value decomposition incorporating stability selection*, Bioinformatics 27 (2011), no. 15, 2089–2097.

15. A. A. Shabalin, V. J. Weigman, C. M. Perou, and A. B. Nobel, *Finding large average submatrices in high dimensional data*, Ann. Appl. Stat. 3 (2009), no. 3, 985–1012.

16. T. S. Ferguson, *A Bayesian analysis of some nonparametric problems*, Ann. Stat. 1 (1973), 209–230.

17. D. J. Aldous, *Exchangeability and related topics*. École d’Été St Flour 1983, Lecture Notes in Math. 1117, Springer-Verlag, Berlin, Germany, 1985, 1–198.

18. D. Blackwell and J. B. MacQueen, *Ferguson distributions via Pólya urn schemes*, Ann. Stat. 1 (1973), no. 2, 353–355.

19. A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, *Bayesian Data Analysis*, 3rd ed., Chapman and Hall/CRC, New York, 2013.

20. W. M. Rand, *Objective criteria for the evaluation of clustering methods*, J. Am. Stat. Assoc. 66 (1971), no. 336, 846–850.

21. J. Geweke, “Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments,” *Bayesian Statistics*, 4, Clarendon Press, Oxford, 1992, pp. 169–193.

22. A. Gelman and D. B. Rubin, *Inference from iterative simulation using multiple sequences*, Stat. Sci. 7 (1992), no. 4, 457–472.

23. Y. Liu, D. N. Hayes, A. Nobel, and J. S. Marron, *Statistical significance of clustering for high-dimension, low–sample size data*, J. Am. Stat. Assoc. 103 (2008), no. 483, 1281–1293.

24. T. Sørlie, C. M. Perou, R. Tibshirani, T. Aas, S. Geisler, H. Johnsen, T. Hastie, M. B. Eisen, M. van de Rijn, S. S. Jeffrey, T. Thorlsen, H. Quist, J. C. Matese, P. O. Brown, D. Botstein, P. E. Lønning, and A. L. Berresen-Dale, *Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications*, Proc. Natl. Acad. Sci. 98 (2001), no. 19, 10869–10874.

25. R. W. Tothill, A. V. Tinker, J. George, R. Brown, S. B. Fox, S. Lade, D. S. Johnson, M. K. Trivett, D. Etemadmoghadam, B. Locandro, N. Traficante, S. Fereday, J. A. Hung, Y. E. Chiw, I. Haviv, Australian Ovarian Cancer Study Group, D. Gertig, A. de Fazio, and D. D. L. Bowtell, *Novel molecular subtypes of serous and endometrioid ovarian cancer linked to clinical outcome*, Clin. Cancer Res. 14 (2008), no. 16, 5198–5208.

26. J. Yang, H. Wang, W. Wang, and P. S. Yu, *An improved biclustering method for analyzing gene expression profiles*, Int. J. Artif. Intell. Tools 14 (2005), no. 5, 771–789.

27. J. Gu and J. S. Liu, *Bayesian biclustering of gene expression data*, BMC Genom. 9 (2008), no. 1, S4.

**AUTHOR BIOGRAPHIES**

Abhishek Chakraborty is an Assistant Professor of Statistics in the Department of Mathematics, Statistics, and Computer Science at Lawrence University, Wisconsin. He earned his PhD in Statistics from Iowa State University. His research interests include the development of statistical methodologies for the analysis of complex datasets, statistical learning, data mining, predictive modeling, application of Bayesian methodologies, statistics pedagogy, and history of statistics.

Stephen B. Vardeman is an Emeritus Faculty of Statistics and Industrial Engineering at Iowa State University. He is a Fellow of the American Statistical Association, an Elected Member of the International Statistical Institute, was LAS Kingland Data Analytics Faculty Fellow 2017 through 2019, and was Editor of Technometrics from 1993 through 1995. His professional interests include Statistical Machine Learning, Business and Engineering Analytics, Engineering and Natural Science applications of Statistics, Statistics and Metrology, Directional Data Analysis, Industrial applications, Statistical Education, and the development of new Statistical Theory and Methods.

**SUPPORTING INFORMATION**

Additional supporting information can be found online in the Supporting Information section at the end of this article.

**How to cite this article:** A. Chakraborty, and S. B. Vardeman, *Some Bayesian biclustering methods: Modeling and inference*, Stat. Anal. Data Min.: ASA Data Sci. J. 15 (2022), 413–432. https://doi.org/10.1002/sam.11584

**APPENDIX A**

A.1. A Relationship between the Joint Densities Mentioned in Section 3

In this appendix, we prove that integrating out $\pi$ and $\phi$ from the joint density in (10) results in the joint density in expression (7).
Recall that we consider a $I \times J$ data matrix $Y$ with the $I$-dimensional vector $r$ with entries $r(i)$ in $\{1, 2, \ldots, R\}$ specifying the cluster assignment of all rows and the $J$-dimensional vector $c$ with entries $c(j)$ in $\{1, 2, \ldots, C\}$ specifying the cluster assignment of all columns. The weight vectors $\pi = (\pi_1, \ldots, \pi_R)$ (for rows) and $\phi = (\phi_1, \ldots, \phi_C)$ (for columns) are given symmetric Dirichlet prior distributions with concentration parameters $\alpha/R$ and $\beta/C$ respectively. The elements of $\pi$ and $\phi$ must be positive and must sum to one.

In the joint density (10), the terms involving $\pi$ and $\phi$ are

$$
\left[ \left( \prod_{i=1}^{I} \pi_{r(i)} \right) \left( \prod_{j=1}^{J} \phi_{c(j)} \right) \right] \left[ \left( \frac{\Gamma(\alpha)}{(\Gamma(\alpha/R))^R} \right) \prod_{r=1}^{R} \pi_{r}^{\alpha/R-1} \right]
\times \left[ \left( \frac{\Gamma(\beta)}{(\Gamma(\beta/C))^C} \right) \prod_{c=1}^{C} \phi_{c}^{\beta/C-1} \right].
$$

Note that, since $r(i); i = 1, \ldots, I$ can take values in $\{1, 2, \ldots, R\}$, we can rearrange the terms in the product $\prod_{i=1}^{I} \pi_{r(i)}$ and write it as $\prod_{r=1}^{R} \pi_{r}^{n_r(r)}$ where $n_r(r)$ has been defined in Section 3. Similarly, with $n_c(c)$ defined in Section 3, we have $\prod_{j=1}^{J} \phi_{c(j)} = \prod_{c=1}^{C} \phi_{c}^{n_c(c)}$. Thus, (A1) can be written as

$$
\left[ \left( \frac{\Gamma(\alpha)}{(\Gamma(\alpha/R))^R} \right) \prod_{r=1}^{R} \pi_{r}^{n_r(r)+\alpha/R-1} \right]
\times \left[ \left( \frac{\Gamma(\beta)}{(\Gamma(\beta/C))^C} \right) \prod_{c=1}^{C} \phi_{c}^{n_c(c)+\beta/C-1} \right]
$$

Then, given (A2), integrating out $\pi$ and $\phi$ from joint density (10) evaluates to

$$
\int \ldots \int \left( \frac{\Gamma(\alpha)}{(\Gamma(\alpha/R))^R} \right) \prod_{r=1}^{R} \pi_{r}^{n_r(r)+\alpha/R-1} d\pi_1 \ldots d\pi_{R-1}
\times \int \ldots \int \left( \frac{\Gamma(\beta)}{(\Gamma(\beta/C))^C} \right) \prod_{c=1}^{C} \phi_{c}^{n_c(c)+\beta/C-1} d\phi_1 \ldots d\phi_{C-1}
= \left( \frac{\Gamma(\alpha)}{(\Gamma(\alpha/R))^R} \right) \int \ldots \int \prod_{r=1}^{R} \pi_{r}^{n_r(r)+\alpha/R-1} d\pi_1 \ldots d\pi_{R-1}
\times \int \ldots \int \left( \frac{\Gamma(\beta)}{(\Gamma(\beta/C))^C} \right) \prod_{c=1}^{C} \phi_{c}^{n_c(c)+\beta/C-1} d\phi_1 \ldots d\phi_{C-1}
$$

$$
= \left[ \left( \frac{\Gamma(\alpha)}{(\Gamma(\alpha/R))^R} \right) \prod_{r=1}^{R} \Gamma(\alpha) / \prod_{r=1}^{R} \pi_{r}^{n_r(r) + \alpha/R - 1} \right]
\times \left[ \left( \frac{\Gamma(\beta)}{(\Gamma(\beta/C))^C} \right) \prod_{c=1}^{C} \phi_{c}^{n_c(c) + \beta/C - 1} \right].
$$

Note that, the space of $\pi_1, \pi_2, \ldots, \pi_R$ has $R - 1$ dimensions since $\sum_{r=1}^{R} \pi_{r} = 1 = \pi_R = 1 - \sum_{r=1}^{R-1} \pi_{r}$. Similarly for $\phi_1, \phi_2, \ldots, \phi_C$. (A4) follows from the pdf of the Dirichlet distribution and (A5) follows from the facts that $\sum_{r=1}^{R} n_r(r) = I$ and $\sum_{c=1}^{C} n_c(c) = J$.

Now, for the first term in (A5), we have

$$
\left( \frac{\Gamma(\alpha)}{(\Gamma(\alpha/R))^R} \right) \prod_{r=1}^{R} \Gamma(\alpha) / \prod_{r=1}^{R} \pi_{r}^{n_r(r) + \alpha/R - 1} = \frac{\Gamma(\alpha)}{\prod_{r=1}^{R} \Gamma(\alpha)} \frac{\prod_{r=1}^{R} \Gamma(\alpha)}{\prod_{r=1}^{R} \pi_{r}^{n_r(r) + \alpha/R - 1}}.
$$

Note that, the space of $\pi_1, \pi_2, \ldots, \pi_R$ has $R - 1$ dimensions since $\sum_{r=1}^{R} \pi_{r} = 1 = \pi_R = 1 - \sum_{r=1}^{R-1} \pi_{r}$. Similarly for $\phi_1, \phi_2, \ldots, \phi_C$. (A4) follows from the pdf of the Dirichlet distribution and (A5) follows from the facts that $\sum_{r=1}^{R} n_r(r) = I$ and $\sum_{c=1}^{C} n_c(c) = J$.

Note that,

$$
\frac{\Gamma(\alpha)}{\prod_{r=1}^{R} \Gamma(\alpha/R)} = \frac{\Gamma(\alpha)}{(\Gamma(\alpha))^R} \frac{\prod_{r=1}^{R} \Gamma(\alpha)}{\prod_{r=1}^{R} \pi_{r}^{n_r(r) + \alpha/R - 1}} = \prod_{r=1}^{R} \frac{1}{\alpha + (I - 1)}.
$$

Also, for those $r$ for which $n_r(r) = 0$, we have $\frac{\Gamma(\alpha)}{\prod_{r=1}^{R} \Gamma(\alpha/R)} = 1$. Otherwise, if $n_r(r) > 0$, then

$$
\frac{\Gamma(n_r(r) + \alpha/R)}{\Gamma(\alpha/R)} = \left( \frac{\alpha}{R} + n_r(r) - 1 \right) \left( \frac{\alpha}{R} + n_r(r) - 2 \right) \ldots \left( \frac{\alpha}{R} + 1 \right).
$$

Expressions (A7) and (A8) follow from the representation of Gamma functions in terms of factorials. Combining (A7) and (A8), Equation (A6) reduces to

$$
\left( \prod_{i=1}^{I} \frac{1}{\alpha + (I - 1)} \right) \prod_{r=1}^{R} \Gamma(\alpha) \frac{\prod_{r=1}^{R} \Gamma(\alpha)}{\prod_{r=1}^{R} \pi_{r}^{n_r(r) + \alpha/R - 1}} \times \left( \frac{\alpha}{R} + n_r(r) - 2 \right) \ldots \left( \frac{\alpha}{R} + 1 \right) = g(r).
$$

where $g(r)$ is defined in display (3). Similarly,

$$
\left( \frac{\Gamma(\beta)}{(\Gamma(\beta/C))^C} \right) \prod_{c=1}^{C} \Gamma(\beta) / \prod_{c=1}^{C} \phi_{c}^{n_c(c) + \beta/C - 1} = h(c)
$$

where $h(c)$ is defined in display (5).
It also follows from rearrangement of terms that,

\[
\begin{align*}
&\left[ \prod_{i=1}^{I} \prod_{j=1}^{J} f(y_{ij} | \mu_{ri}, \sigma^2) \right] \left[ \prod_{r=1}^{R} \prod_{c=1}^{C} f(\mu_{rc} | \mu, \rho^2) \right] \\
&= \prod_{r=1}^{R} \prod_{c=1}^{C} \left( \prod_{(i,j) \in (r, c)} f(y_{ij} | \mu_{rc}, \sigma^2) \right) f(\mu_{rc} | \mu, \rho^2)
\end{align*}
\]

(A11)

Combining displays (A9), (A10), and (A11), we have that integrating out \( \pi \) and \( \phi \) from the joint density (10) results in the joint density (7).