Bayesian Overlapping Subspace Clustering

Qiang Fu
Dept. of Computer Science & Engineering
University of Minnesota, Twin Cities
qifu@cs.umn.edu

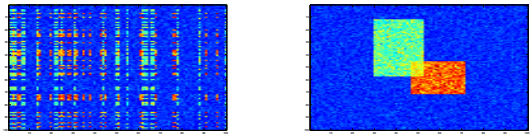
Arindam Banerjee
Dept. of Computer Science & Engineering
University of Minnesota, Twin Cities
banerjee@cs.umn.edu

Abstract

Given a data matrix, the problem of finding dense/uniform sub-blocks in the matrix is becoming important in several applications. The problem is inherently combinatorial since the uniform sub-blocks may involve arbitrary subsets of rows and columns and may even be overlapping. While there are a few existing methods based on co-clustering or subspace clustering, they typically rely on local search heuristics and in general do not have a systematic model for such data. We present a Bayesian Overlapping Subspace Clustering (BOSC) model which is a hierarchical generative model for matrices with potentially overlapping uniform sub-block structures. The BOSC model can also handle matrices with missing entries. We propose an EM-style algorithm based on approximate inference using Gibbs sampling and parameter estimation using coordinate descent for the BOSC model. Through experiments on both simulated and real datasets, we demonstrate that the proposed algorithm outperforms the state-of-the-art.

1 Introduction

Several datasets are represented in the form of a matrix, where each row represents an object and each column represents a feature. The problem of finding dense/uniform sub-blocks in a given data matrix has emerged as an important unsupervised learning task. A dense/uniform sub-block consists of a subset of instances that have similar feature values for a subset of features. Such a problem is inherently combinatorial and has been investigated in the context of subspace clustering [2, 15], projected clustering [1, 18] and co-clustering [8, 11, 17]. The problem is important in a variety of applications. For example, in gene expression data analysis, one would like to find a set of genes which co-express under a set of experimental conditions. In a recommendation system, a uniform sub-block indicates a group of users who have similar ratings for a group of movies.



(a) Raw Data (b) Ideal Output

Figure 1. An example problem: (a) Raw data with latent overlapping co-clustering structure, (b) Ideal output from an algorithm, where rows and columns have been permuted to reveal the structure discovered.

While progress has been made in the development of subspace clustering and co-clustering algorithms, the existing formulations often lack the flexibility needed to solve the problem of finding uniform sub-blocks. In the current context, the desiderata can be captured by the following three requirements. First, *the sub-blocks may overlap*, so that some entries may belong to more than one sub-block. For example, in gene expression analysis, a gene can have multiple functions and hence co-express with different groups under different experimental conditions. Most clustering/co-clustering formulations are not designed to discover overlapping clusters. Second, *not all rows and columns may be a part of a sub-block*, and the formulation has to be flexible enough to allow that. Most existing clustering/co-clustering formulations assume that all points belong to some cluster/co-cluster, and the corresponding algorithms have no capacity to identify background noise automatically. Finally, *the matrix may have missing entries*. In practice, one often imputes the missing values with row/column statistics or has a heuristic work around. Ideally, we want the model formulation to be able to work with sparse matrices and in fact use sparsity to a computational advantage. To better understand the task the algorithm should perform, we give an example in Figure 1. Figure 1(a) shows a simple input data matrix with two overlapping dense blocks and background noise and Figure 1(b) is the ideal output of the algorithm.

In this paper, we present a BOSC model which can find potentially overlapping sub-blocks, automatically de-

tect background noise and naturally handle matrices with missing entries. For inference and parameter estimation, we propose an EM-style algorithm.

The rest of the paper is organized as follows: We propose the BOSC model in Section 2 and present an EM-style algorithm to learn the sub-block assignments in Section 3. The experimental results on both simulated and real datasets are presented in Section 4. We conclude in Section 5.

2 Bayesian Overlapping Subspace Clustering Model

The proposed Bayesian Overlapping Subspace Clustering model assumes that the number of sub-blocks k is given. Each sub-block is modeled using a parametric distribution $p(\cdot|\boldsymbol{\theta}_j), [j]_1^k$ $([j]_1^k \equiv j = 1, \ldots, k)$ from any suitable exponential family. The noise entries are modeled using another distribution $p(\cdot|\boldsymbol{\theta}_{k+1})$ from the same family. However, the generative model for the observed data matrix is rather different from traditional mixture models [12] as well as the more recent mixed membership models such as LDA [3, 7].

Suppose the data matrix X has m rows and n columns, possibly with several missing entries. The main idea behind the proposed model is as follows: Each row u and each column v respectively have k-dimensional latent bit vectors $\mathbf{z}_r^u$ and $\mathbf{z}_c^v$ which indicate their sub-block memberships. The sub-block membership for any entry x_{uv} in the matrix is obtained by an element-wise (Hadamard) product of the corresponding row and column bit vectors, i.e., $\mathbf{z} = \mathbf{z}_r^u \odot \mathbf{z}_c^v$. Given the sub-block membership $\mathbf{z}$ and the sub-block distributions, the actual observation x_{uv} is assumed to be generated by a multiplicative mixture model [9, 4] so that

$$p(x_{uv}|\mathbf{z}_r^u, \mathbf{z}_c^v, \Theta) = \begin{cases} \frac{1}{c(\mathbf{z})} \prod_{j=1}^k p_j(x_{uv}|\boldsymbol{\theta}_j)^{z_j} & \text{if } \mathbf{z} \neq \mathbf{0}\ , \\ p(x_{uv}|\boldsymbol{\theta}_{k+1}) & \text{otherwise}\ , \end{cases} \tag{1}$$

where $c(\mathbf{z})$ is a normalization factor to guarantee that $p(\cdot|\mathbf{z}_r^u, \mathbf{z}_c^v, \Theta)$ is a valid distribution. If $\mathbf{z} = \mathbf{z}_r^u \odot \mathbf{z}_c^v = \mathbf{0}$, the all zeros vector, then x_{uv} is assumed to be generated from the noise component $p(\cdot|\boldsymbol{\theta}_{k+1})$. In the sequel, we will use $[\mathbf{z}_r^u \odot \mathbf{z}_c^v = \mathbf{0}]$ to denote the indicator of this event. Figure 1 shows an example of a matrix generated from such a model with two dense blocks. The Hadamard product ensures that the matrix has uniform/dense sub-blocks with possible overlaps while treating certain rows/columns as noise.

Since it can be tricky to work directly with latent bit vectors, we introduce suitable Bayesian priors on the sub-block memberships. In particular, the proposed model assumes that there are k Beta distributions $\mathrm{Beta}(\alpha_r^j, \beta_r^j), [j]_1^k$ corresponding to the rows and k Beta distributions $\mathrm{Beta}(\alpha_c^j, \beta_c^j), [j]_1^k$ corresponding to the columns. Let $\pi_r^{u,j}$ denote the Bernoulli parameter sampled from $\mathrm{Beta}(\alpha_r^j, \beta_r^j)$ for row u and sub-block j where $[u]_1^m$ and $[j]_1^k$. Similarly, let $\pi_c^{v,j}$ denote the Bernoulli parameter sampled from $\mathrm{Beta}(\alpha_c^j, \beta_c^j)$ for column v and sub-block j, where $[v]_1^n$ and $[j]_1^k$. The Beta-Bernoulli distributions are assumed to be the priors for the latent row and column membership vectors $\mathbf{z}_r^u$ and $\mathbf{z}_c^v$.

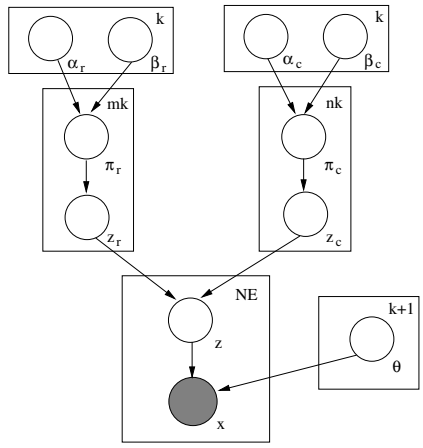


Figure 2. Bayesian Overlapping Subspace Clustering model. $\mathbf{z} = \mathbf{z}_r \odot \mathbf{z}_c$. NE is the number of non-missing entries in the matrix.

The proposed model is shown as a plate diagram in Figure 2. In particular, the generative process is as follows:

1. For each co-cluster $[j]_1^k$:
   (a) For each row $u, [u]_1^m$:
      (i) sample $\pi_r^{u,j} \sim \mathrm{Beta}(\alpha_r^j, \beta_r^j)$,
      (ii) sample $z_r^{u,j} \sim \mathrm{Bernoulli}(\pi_r^{u,j})$.
   (b) For each column $v, [v]_1^n$:
      (i) sample $\pi_c^{v,j} \sim \mathrm{Beta}(\alpha_c^j, \beta_c^j)$,
      (ii) sample $z_c^{v,j} \sim \mathrm{Bernoulli}(\pi_c^{v,j})$.
2. For each (non-missing) matrix entry $x_{uv}, [u]_1^m [v]_1^n$, sample

$$x_{uv} \sim \begin{cases} \frac{1}{c(\mathbf{z}_r^u \odot \mathbf{z}_c^v)} \prod_{j=1}^k p(x_{u,v}|\theta_j, \mathbf{z}_r^{u,j}, \mathbf{z}_c^{v,j}) & \text{if } z_r^u \odot \mathbf{z}_c^v \neq \mathbf{0}\ , \\ p(x_{u,v}|\theta_{k+1}) & \text{otherwise}\ . \end{cases}$$

Since only the observed entries in the matrix are assumed to be generated by the above process, the model naturally handles matrices with missing values.

3 Analysis and Algorithm

Let Π_r and Π_c be $m \times k$ and $n \times k$ latent matrices which have the Bernoulli parameters for each row and column, Z_r and Z_c be $m \times k$ and $n \times k$ binary matrices that have the latent row and column sub-block assignments for each row and column. For convenience of notation, let ς_{uv} be an

indicator variable for observed entries in the matrix, i.e., $\varsigma_{uv} = 1$ if entry x_{uv} is not missing, and 0 otherwise. Then the joint distribution over all observed and latent variables is given by

$$\begin{aligned} &p(X, Z_r, Z_c, \Pi_r, \Pi_c | \boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta) \\ &\quad = p(\Pi_r | \boldsymbol{\alpha}_r, \boldsymbol{\beta}_r) p(\Pi_c | \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c) p(Z_r | \Pi_r) p(Z_c | \Pi_c) \\ &\qquad p(X | \Theta, Z_r, Z_c) \ . \end{aligned} \tag{2}$$

Since the observations are statistically independent given Z_r, Z_c, we have

$$p(X | \Theta, Z_r, Z_c) = \prod_{u=1}^{m} \prod_{v=1}^{n} p(x_{uv} | \Theta, \mathbf{z}_r^u, \mathbf{z}_c^v)^{\varsigma_{uv}} \ . \tag{3}$$

Marginalizing over all latent variables, the conditional probability of generating the matrix X given the parameters $(\boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta)$ is given by

$$\begin{aligned} &p(X | \boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta) \\ &= \int_{\Pi_r, \Pi_c} \sum_{Z_r, Z_c} p(X, Z_r, Z_c, \Pi_r, \Pi_c | \boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta) d\Pi_r d\Pi_c \ . \end{aligned} \tag{4}$$

Π_r and Π_c can be analytically integrated out in (4) because of conjugacy: they are generated by Beta distributions which are conjugate priors to Bernoulli distributions which generate Z_r and Z_c. Thus (4) does not depend on Π_r and Π_c. It is also important to note the conditional probability of observing X as in (4) is not the product of the conditional probability of observing each entry, i.e.,

$$\begin{aligned} &p(X | \boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta) \\ &\qquad \neq \prod_{u=1}^{m} \prod_{v=1}^{n} p(x_{u,v} | \boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta)^{\varsigma_{uv}} \ . \end{aligned} \tag{5}$$

The equality does not hold because the entries in the matrix are not conditionally independent given the parameters $(\boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta)$. According to the generative process, $\mathbf{z}_r^u$ and $\mathbf{z}_c^v$ are sampled only once for each row and column, so that the observations in the same row/column get coupled. This is a crucial departure from several related mixture models which assume the joint probability of all observations to be simply a product of the marginal probabilities.

Given the entire matrix X, the learning task is to infer the joint posterior distribution of (Z_r, Z_c) and compute the model parameters $(\boldsymbol{\alpha}_r^\star, \boldsymbol{\beta}_r^\star, \boldsymbol{\alpha}_c^\star, \boldsymbol{\beta}_c^\star, \Theta^\star)$ which maximize $\log p(X | \boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta)$. We can then draw samples from the posterior distribution and compute the dense-block assignment for each entry. A general approach for the learning task is to use expectation maximization (EM) algorithm [13]. However, direct calculation of $\log p(X | \boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta)$ is intractable, indicating that a direct application of EM is not possible. In this section, we propose an EM-like algorithm alternating between approximate inference and optimal parameter estimation to tackle the learning task.

3.1 Inference

In the E-step, given the model parameters $(\boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta)$, the goal is to estimate the expectation of the log-likelihood $E[\log p(X, Z_r, Z_c | \boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta)]$ where the expectation is with respect to the posterior probability $p(Z_r, Z_c | X, \boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta)$. We use Gibbs sampling to approximate the expectation [7, 5]. In particular, we compute the conditional probabilities of each row variable $z_r^{u,j}$ and column variable $z_c^{v,j}$ and construct a Markov chain based on the conditional probabilities. On convergence, the chain will draw samples from the posterior joint distribution of (Z_r, Z_c), which in turn can be used to get an approximate estimate of the expected log-likelihood.

If $Z_r^{-(u,j)}$ denotes the binary matrix Z_r excluding $z_r^{u,j}$, the conditional probability of $z_r^{u,j} = 1$ is given by:

$$\begin{aligned} &p(z_r^{u,j} = 1 | Z_r^{-(u,j)}, Z_c, X, \Theta) \\ &\quad \propto p(X | Z_r, Z_c, \Theta) p(z_r^{u,j} = 1 | Z_r^{-(u,j)}) \ , \end{aligned}$$

where $p(X | Z_r, Z_c, \Theta)$ is as in (3) and

$$\begin{aligned} p(z_r^{u,j} = 1 | Z_r^{-(u,j)}) &= \int_0^1 p(z_r^{u,j} = 1 | \pi_r^{u,j}) p(\pi_r^{u,j}) d\pi_r^{u,j} \\ &= \frac{\alpha_r^j}{\alpha_r^j + \beta_r^j} \ . \end{aligned} \tag{6}$$

Now,

$$\begin{aligned} &p(z_r^{u,j} = 1 | Z_r^{-(u,j)}, Z_c, X, \Theta) \\ &\propto \prod_{p=1}^{m} \prod_{q=1}^{n} p(x_{p,q} | \mathbf{z}_r^p, \mathbf{z}_c^q, \Theta)^{\varsigma_{pq}} . \frac{\alpha_r^j}{\alpha_r^j + \beta_r^j} \ , \end{aligned} \tag{7}$$

$$\propto \prod_{q=1}^{n} p(x_{u,q} | \mathbf{z}_r^u, \mathbf{z}_c^q, \Theta)^{\varsigma_{uq}} . \frac{\alpha_r^j}{\alpha_r^j + \beta_r^j} \ , \tag{8}$$

$$\propto \prod_{q=1}^{n} \left(\frac{p(x_{u,q} | \boldsymbol{\theta}_j)^{z_c^{q,j}} . p(x_{u,q} | \boldsymbol{\theta}_{k+1})^{[\mathbf{z}_r^u \odot \mathbf{z}_c^q = \mathbf{0}]}}{c(\mathbf{z}_r^u \odot \mathbf{z}_c^q)} \right)^{\varsigma_{uq}} . \frac{\alpha_r^j}{\alpha_r^j + \beta_r^j} \ , \tag{9}$$

where (8) follows since the probability of generating the entries in the rows except u does not depend on the value of $z_r^{u,j}$, and (9) follows since whether the entry $x_{u,q}$ belongs to sub-blocks other than j does not play a role in deciding the value of $z_r^{u,j}$ in the product term other than the overall normalization term $c(\mathbf{z}_r^u \odot \mathbf{z}_c^q)$.

The probability of $z_r^{u,j} = 0$ can be derived similarly as

$$\begin{aligned} &p(z_r^{u,j} = 0 | Z_r^{-(u,j)}, Z_c, X, \Theta) \\ &\quad \propto \prod_{q=1}^{n} \left(\frac{p(x_{u,q} | \boldsymbol{\theta}_{k+1})^{[\mathbf{z}_r^u \odot \mathbf{z}_c^q = \mathbf{0}]}}{c(\mathbf{z}_r^u \odot \mathbf{z}_c^q)} \right)^{\varsigma_{uq}} . \frac{\beta_r^j}{\alpha_r^j + \beta_r^j} \ . \end{aligned} \tag{10}$$

The conditional probabilities for the other binary assignment variables can be similarly derived.

The true underlying posterior distribution of (Z_r, Z_c) may have multiple modes. To prevent the sampling algorithm from getting stuck in local modes, we modify the Gibbs sampler using simulated annealing [10]. Given a temperature parameter T, the sampling is done following

$$p^{(T)}(z_r^{u,j} = 0|\cdots) = \frac{p(z_r^{u,j} = 0|\cdots)^{\frac{1}{T}}}{p(z_r^{u,j} = 0|\cdots)^{\frac{1}{T}} + p(z_r^{u,j} = 1|\cdots)^{\frac{1}{T}}},$$

$$p^{(T)}(z_r^{u,j} = 1|\cdots) = \frac{p(z_r^{u,j} = 1|\cdots)^{\frac{1}{T}}}{p(z_r^{u,j} = 0|\cdots)^{\frac{1}{T}} + p(z_r^{u,j} = 1|\cdots)^{\frac{1}{T}}}.$$

When T is high, the probability distribution is almost uniform, and when T is low, more emphasis is given to high probability states. In practice, we start with a relatively high T and gradually decrease T to 1, when the sampling distribution is exactly the posterior distribution of Z_r and Z_c.

The sampling is run for enough iterations till it converges. Then we sample from the stationary distribution (with suitable gaps) to obtain N independent and identically distributed samples of (Z_r, Z_c), where N is a predefined large number. From the samples, the expectation of the log-likelihood can be empirically estimated as: $\frac{1}{N}\sum_{s=1}^{N} \log p(X, Z_{r,s}, Z_{c,s}|\boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta)$, where $Z_{r,s}$ and $Z_{c,s}$ correspond to the sth samples.

3.2 Estimation

In M-step, we estimate $(\boldsymbol{\alpha}_r^\star, \boldsymbol{\beta}_r^\star, \boldsymbol{\alpha}_c^\star, \boldsymbol{\beta}_c^\star, \Theta^\star)$ which maximizes the expectation. Note that, given Z_r and Z_c, each entry in the matrix is statistically independent of each other. So the parameter estimation problem can be formulated as maximizing the following expected log-likelihood objective function:

$$\begin{aligned}
&L(\boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c, \Theta) \\
&= \sum_{s=1}^{N} \log p(X, Z_{r,s}, Z_{c,s}|\boldsymbol{\alpha}_r, \boldsymbol{\alpha}_c, \Theta) \\
&= \sum_{s=1}^{N} \log p(Z_{r,s}|\boldsymbol{\alpha}_r) + \sum_{s=1}^{N} \log p(Z_{c,s}|\boldsymbol{\alpha}_c) \\
&\quad + \sum_{s=1}^{N} \log p(X|Z_{r,s}, Z_{c,s}, \Theta) \\
&= \sum_{s=1}^{N}\sum_{u=1}^{m}\sum_{j=1}^{k} \log p(z_{r,s}^{u,j}|\alpha_r^j) + \sum_{s=1}^{N}\sum_{v=1}^{n}\sum_{j=1}^{k} \log p(z_{c,s}^{v,j}|\alpha_c^j) \\
&\quad + \sum_{s=1}^{N}\sum_{u=1}^{m}\sum_{v=1}^{n} \varsigma_{uv} \log p(x_{u,v}|\mathbf{z}_{r,s}^u, \mathbf{z}_{c,s}^v, \Theta) \,. \qquad (11)
\end{aligned}$$

The optimization can be broken into two independent parts—over the parameters $(\boldsymbol{\alpha}_r, \boldsymbol{\beta}_r, \boldsymbol{\alpha}_c, \boldsymbol{\beta}_c)$ of the Beta distributions, and over the natural parameters Θ of the exponential family distributions. The parameter update method is similar to the one in [4]. Due to the space constraint, we omit the details.

4 Experimental Results

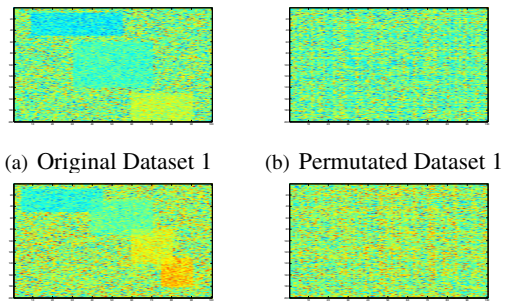
In this section, we present experimental results on simulated datasets, a microarray gene expression dataset and a movie recommendation dataset. First, we introduce some additional notation to be used in this section: T_{start} denotes the initial temperature parameter in simulated annealing, $f_T < 1$ denotes the multiplicative factor by which the temperature goes down every I_T iterations and N is the number of samples drawn from the stationary distribution.

Since we obtain several samples from the Markov chain after it converges, the final row and column sub-block assignments are decided by the following appraoch: if a row/column belongs to a sub-block in more than half of the samples, we consider the row/column belongs to that corresponding sub-block.

4.1 Simulated Datasets

We do experiment on four simulated datasets. The first two datasets are easy to visualize and both datasets are in the form of a 200×100 matrix, whose entries are initially sampled from a background noise distribution. For the first dataset D_1, we introduce 3 non-overlapping uniform blocks (normally distributed with different means) to replace certain sub-blocks in the matrix (Figure 3(a)). The actual dataset is obtained by randomly permuting the rows and columns, so that the block structure is not apparent (Figure 3(b)). For the second dataset D_2, we introduce 4 mildly overlapping dense blocks where the overlapping entries are generated from the multiplicative model in (1) (Figure 3(c)). As before, the actual dataset is obtained by a random row/column permutation (Figure 3(d)). The other two simulated datasets have larger number of sub-blocks, one with 10 blocks and the other with 15 blocks. We do not provide label information to STATPC on these two datasets.

We compare the performance of the proposed algorithm to a state-of-the-art subspace clustering algorithm called STATPC [15] and an overlapping clustering algorithm [4], which we call MMM. STATPC finds non-redundant and statistically (overlapping) regions in high dimensional data. MMM finds overlapping clusters and automatically detects the noisy data points. To make the three algorithms comparable, MMM is used to cluster the entries in the matrix, instead of the rows. STATPC can make use of the row cluster labels if available—we give it substantial advantage by providing the true cluster labels for all the rows. For D_1,



(a) Original Dataset 1 (b) Permutated Dataset 1

(c) Original Dataset 2 (d) Permutated Dataset 2

Figure 3. Two Simulation Datasets.

Accuracy	Dataset 1	Dataset 2	Dataset 3	Dataset 4
BOSC	**1**	**0.8959**	**0.6813**	**0.5723**
STATPC	0.7767	0.4786	0.2670	0.1286
MMM	0.5888	0.5287	0.4560	0.3549

Table 1. Clustering accuracy of BOSC, STATPC and MMM on four simulation datasets.

since there is no overlap, we provide the true cluster label. For D_2, we provide the true cluster labels for the non-overlapping rows, and one of the true cluster labels for the overlapping rows. On the contrary, the proposed algorithm does not use any form of supervision. In particular, we run kmeans on the matrix entries to get the initial estimate of the component parameter values for BOSC and MMM—in this case, means and standard deviations of each Gaussian component. However, kmeans does not capture the structure of the matrix, because it rarely assigns entries to the correct sub-blocks. The noise component is initialized with the mean and standard deviation across all entries in the matrix. We use $T_{start} = 10$, $f_T = 0.67$, $I_T = 50$ and $N = 50$.

We quantitatively measure the performance by calculating the clustering accuracy (Table 1). In particular, suppose c_1 is the number of ground truth sub-blocks, c_2 is the number of output sub-blocks, and a_{ij}, $[i]_1^{c_1}[j]_1^{c2}$ is the number of entries from the ith ground truth block that are also in the jth output block. The clustering accuracy is defined as :

$$\text{Clustering Accuracy} = \frac{\sum_{j=1}^{c2} \max_i a_{ij}}{\sum_{i=1}^{c1} \sum_{j=1}^{c2} a_{ij}} .$$

4.2 Microarray Gene Expression Dataset

The microarray gene expression dataset we use consists of 4062 genes and 215 experimental conditions [14]. We first select 1000 genes that have the highest variance of expressions over the 215 conditions. We run our algorithm on this 1000×215 matrix and want to find subsets of genes which highly co-express under subsets of conditions. The number of sub-blocks is set to be 30. The annealing parameters are set as follows: $T_{start} = 500$, $f_T = 0.67$, $I_T = 75$ and $N = 100$. As a strong baseline, we use the the Plaid bi-clustering algorithm [11] which has been extensively used for gene-expression analysis. The Plaid algorithm finds overlapping dense regions in gene-expression datasets. We also compare our algorithm with a model-based overlapping co-clustering (MOC) algorithm [16] and the state-of-the-art Bayesian co-clustering (BOC) algorithm [17].

Algorithms	BOSC	Plaid	MOC	BOC
Number of Blocks with Significant Enrichment	13	8	10	10

Table 2. BOSC finds more dense blocks with significant enrichment.

To evaluate whether the dense blocks identified are meaningful from a biological perspective, we check if the genes in each dense block show significant enrichment for known biological process annotations. We make use of Gene Ontology Term Finder[1] online tool, which searches for shared annotations given a set of genes and computes an associated p-value. The p-value measures the probability of observing a group of genes to be annotated with a certain annotation purely by chance. If genes in a dense block indeed correspond to known biological processes, we would expect a low p-value. We consider an annotation to be significant if the p-value associated with it is less than 10^{-4}.

We initialize all algorithms by running kmeans on matrix entries. For co-clustering algorithms, we try different combinations of row/colum cluster numbers and report the best results. The BOC algorithm estimates for each row/column the probability of belonging to each row/column cluster. We consider that a row/column belongs to a row/column cluster if it has the highest probability on that row/column cluster.

The result is listed in Table 2. BOSC identifies 24 blocks of which 13 have significant enrichment. Most of the other identified blocks have p-values that are of the order of 10^{-4}. In contrast, among the 30 'layers' found by Plaid, only 8 have significant enrichment. Among the 30 co-clusters found by MOC, 10 have significant enrichment. BOC also finds 10 blocks with significant enrichment.

4.3 MovieLens Dataset

MovieLens[2] is a movie rating dataset with 100,000 ratings from 943 users on 1682 movies. The ratings are on a 1-5 scale. We work with a subset with 568 users who submitted at least 50 ratings and 603 movies which have at least 50 ratings. The resulting matrix has 73544 ratings and 79% missing entries. Since different users may have different standards and ratings can be very personal, we z-score the ratings submitted by each user.

If we treat each genre as a cluster, the MovieLens dataset has naturally overlapping cluster structure, because each

[1] http://www.yeastgenome.org/cgi-bin/GO/goTermFinder.pl

[2] http://www.grouplens.org/taxonomy/term/14

Dataset1	BOSC	BOC
Precision	0.7722	**0.7727**
Recall	**0.6050**	0.3335
F-measure	**0.6784**	0.4659

Table 3. The performance of BOSC and BOC on the first dataset with animation, children's and comedy movies.

Dataset2	BOSC	BOC
Precision	0.6496	**0.6643**
Recall	**1**	0.5567
F-measure	**0.7876**	0.6058

Table 4. The performance of BOSC and BOC on the second dataset with thriller, action and adventure movies.

movie may have several genres. Following the methodology in [16], we then cerate 2 subsets from the dataset we use above. The first dataset contains 220 movies from 3 genres: animation, children's and comedy. The second dataset contains 225 movies from 3 genres: thriller, action and adventure. We run the BOSC algorithm with $k = 20$ on these 2 datasets trying to discover genres based on the belief that similarity in the user ratings gives an indication about whether the movies are in related genres. Since the BOC algorithm [17] can handle datasets with missing entries, we use it as the baseline. We initialize two algorithms by running kmeans on the matrix entries. The annealing parameters are the same as those in the gene expression experiment. We compare pairwise precision, recall and F-measure over movies. Two movies are in a pair if they belong to the same cluster/genre. Precision, recall and F-measure are calculated as follows:

$$\text{Precision} = \frac{\text{Number of Correctly Identified Pairs}}{\text{Number of Identified Pairs}},$$

$$\text{Recall} = \frac{\text{Number of Correctly Identified Pairs}}{\text{Number of Pairs in the Original Dataset}},$$

$$\text{F-measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

For the BOC algorithm, we try different combinations of the number of row clusters and column clusters and report the best result. The result is listed in Table 3 and 4. Two algorithms have comparable performance on precision, but BOSC has higher recall and F-score.

5 Conclusions

In this paper, we have presented a systematic hierarchical generative model and corresponding algorithms for discovering uniform sub-blocks in a given data matrix. Preliminary empirical evidence goes significantly in favor of the proposed algorithm. Perhaps more importantly, the BOSC model introduces a substantially novel way to approach the problem. There are at least two important future research directions. First, the BOSC model assumes that the number of co-clusters k is known, which is typically not the case in several problems. We would like to investigate nonparametric priors, such as the Indian Buffet Process [6], to relax this assumption. Second, in spite of the effectiveness of the proposed algorithm, it is computationally slow for large datasets. In future, we would like to investigate faster approximate inference algorithms for the BOSC model.

Acknowledgements: The research was supported by NSF grant IIS-0812183.

References

[1] C. C. Aggarwal, J. L. Wolf, P. S. Yu, C. M. Procopiuc, and J. S. Park. Fast algorithms for projected clustering. *SIGMOD*, 1999.

[2] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. *SIGMOD*, 1998.

[3] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[4] Q. Fu and A. Banerjee. Multiplicative mixture models for overlapping clustering. *ICDM*, 2008.

[5] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *PAMI*, 6:721–741, 1984.

[6] T. Griffiths and Z. Ghahramani. Infinite latent feature models and the Indian buffet process. *NIPS*, 2005

[7] T. L. Griffiths and M. Steyvers. Finding scientific topics. *PNAS*, 101(1):5228–5225, 2004.

[8] J. A. Hartigan. Direct clustering of a data matrix. *JASA*, 67(337):123–129, 1972.

[9] K. A. Heller and Z. Ghahramani. A nonparametric bayesian approach to modeling overlapping clusters. *AISTATS*, 2007.

[10] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220:671–680, 1983.

[11] L. Lazzeroni and A. Owen. Plaid models for gene expression data. *Statistica Sinica*, 12:61–86, 2002.

[12] G. Mclachlan and D. Peel. *Finite Mixture Models*. Wiley Series in Probability and Statistics. Wiley-Interscience, 2000.

[13] G. J. McLachlan and T. Krishnan. *The EM algorithm and Extensions*. Wiley-Interscience, 1996.

[14] S. Mnaimneh, A. P. Davierwala, J. Haynes, J. Moffat, W.-T. Peng, W. Zhang, X. Yang, J. Pootoolal, G. Chua, and A. Lopez. Exploration of essential gene functions via titratable promoter alleles. *Cell*, 118(1):31–44, 2004.

[15] G. Moise and J. Sander. Finding non-redundant, statistically significant regions in high dimensional data: a novel approach to projected and subspace clustering. *SIGKDD*, 2008.

[16] M. Shafie and E. Milios. Model-based overlapping co-clustering. *SDM*, 2006.

[17] H. Shan and A. Banerjee. Bayesian co-clustering. *ICDM*, 2008.

[18] K. Y. Yip, D. W. Cheung, and M. K. Ng. Harp: A practical projected clustering algorithm. *TKDE*, 16(11):1387–1397, 2004.