

# Bregman Bubble Co-clustering

Meghana Deodhar    Hyuk Cho    Gunjan Gupta    Joydeep Ghosh    Inderjit Dhillon

{deodhar/ghosh}@ece.utexas.edu  
{hyukcho/inderjit}@cs.utexas.edu  
{gunjang}@amazon.com

IDEAL-2007-09\*

**Intelligent Data Exploration & Analysis Laboratory**

( Web: <http://www.ideal.ece.utexas.edu/> )

Department of Electrical and Computer Engineering  
The University of Texas at Austin  
Austin, Texas 78712  
U.S.A.

October 1, 2007

## Abstract

Clustering problems often involve datasets where only a part of the data is relevant to the problem e.g. in microarray data analysis only a subset of the genes show interesting patterns within a subset of the conditions(features). On such datasets, in order to accurately identify meaningful clusters, the non-informative data points should be automatically detected and pruned and non-discriminative features should be simultaneously discarded. Additionally, since clusters could exist in different subspaces of the feature space, a clustering algorithm that is capable of identifying clusters along multiple axes is more suitable as compared to one that is restricted to traditional “one-sided” clustering. We propose Bregman Bubble co-clustering (BBCC), an approach that generalizes both, Bregman bubble clustering [GG06] and Bregman co-clustering [BDG<sup>+</sup>07] to a scalable and very versatile framework. BBCC works with a large variety of distance measures and different co-cluster definitions, making it applicable to a wide range of real life datasets. We also provide insights into a soft version of our algorithm and the underlying generative model. We further extend BBCC to address the problem of efficiently detecting arbitrarily positioned, possibly overlapping co-clusters in a dataset and combine it with a novel model selection strategy that also automatically determines the appropriate number of co-clusters. We highlight the effectiveness of our approach through extensive experimentation on synthetic as well as real datasets. We show that BBCC not only performs better than traditional approaches on microarray datasets but even improves upon human curated techniques in a completely unsupervised manner.

## 1 Motivation

When clustering certain real world datasets, it has often been observed that only a part of the data forms cohesive clusters. For example in the case of microarray data, typically only a small subset of the genes show coherent and interesting patterns and the rest can be considered non-informative [GG06]. Moreover, for such data, coherent clusters could exist in subspaces formed by different subsets of features [ARD06] e.g. different subsets of genes may be correlated across different subsets of experiments. Additionally, it is possible that some features are not relevant with respect to any cluster.

Traditional clustering algorithms like k-means do not address either issue and assign every data point to a cluster based on a similarity measure computed across all the features. Clustering can be preceded by a feature selection step to retain a discriminative subset of the features, but this still does not allow clusters existing in different subsets of the feature space to be detected easily. Subspace clustering is an approach that combines feature selection with clustering and can find clusters in different, possibly overlapping subspaces, using a “search and evaluate” technique. Subspace clustering algorithms are designed for clustering high dimensional datasets where noisy and irrelevant dimensions can mask existing clusters. However, most of these algorithms have limited utility as they are very expensive, do not scale well and need extensive tuning to get meaningful results. In this paper, we propose Bregman bubble co-clustering, a very general and efficient framework that addresses both these issues simultaneously and effectively.

## 2 Background

The Bregman Bubble Clustering (BBC) technique proposed by Gupta and Ghosh [GG06] addresses the problem of discovering multiple, dense and coherent regions in a dataset while discarding the relatively non-coherent parts of the data. BBC provides a robust, scalable framework for clustering only a relevant fraction of the data. However, this framework was developed for one-sided clustering only, where the data points are clustered based on their similarity across the entire set of features.

Co-clustering, also known as biclustering, is a technique that simultaneously clusters the data along multiple axes, e.g. in the case of microarray data it simultaneously clusters the genes as well as the experiments [CC00, CDGS04]. Co-clustering hence exploits the duality between the data points and the features to improve on one-sided clustering. Since co-clustering measures similarity among data points across different subsets of features, it can detect clusters existing in different subspaces of the feature space. Bregman Co-clustering (BCC), proposed by Banerjee et al. [BDG<sup>+</sup>07] is a very efficient, generalized framework for partitional co-clustering [MO04] that works with any distance measure that is a Bregman divergence. BCC can also be thought of as a matrix approximation technique that approximates the data matrix  $Z$  by a matrix  $\hat{Z}$ , which is constructed such that it preserves a certain set of sufficient statistics within each co-cluster. Banerjee et al. identify 6 possible sets of summary statistics, of increasing complexity, that one might be interested in preserving in the reconstructed matrix  $\hat{Z}$ . These 6 sets of statistics lead to 6 different approximation schemes referred to as co-clustering bases. For example, basis 2 approximates all the values within a co-cluster by the mean of the values and hence preserves the co-cluster means.

The Bregman co-clustering framework is however restricted to partitional co-clustering and assigns every point in the data matrix to exactly one co-cluster i.e. the co-clustering is exhaustive and exclusive. All the data points and features are hence considered useful for the clustering task, which might not always be the case in real datasets. We want an algorithm that will automatically detect and prune away non-informative or “outlier” data points and also perform feature selection by eliminating non-discriminative features. Assuming that the number of relevant data points and features is known, our objective is to identify these relevant data points and features from the entire dataset and simultaneously cluster only the relevant data into coherent co-clusters.

### 3 Contributions

In this paper, we propose Bregman Bubble Co-clustering (BBCC), a generalization of BBC and BCC that aims at finding coherent co-clusters in a dataset while retaining only a fraction of the data points and/or features. Like BCC, BBCC finds co-clusters arranged in a grid structure, but only a predetermined number of rows and columns are assigned to the co-clusters. Figure 1 illustrates the nature of the clusters/co-clusters identified by BBC, BCC and BBCC respectively. BBCC works with all the six possible BCC schemes, each one maintaining a different set of co-cluster statistics. BBCC is a natural extension of both BBC and BCC that combines their capabilities, giving a very general framework applicable to a wide range of real life datasets. For example, in case of microarray data analysis BBCC can be used to find subsets of genes with similar expression patterns across subsets of experiments, while simultaneously discarding parts of the data that show no interesting patterns. Another example is that of mining market basket data, which can be viewed as a matrix of customer-product purchases. This data is very sparse and often only a small number of customer and product combinations show coherent trends in customer preference. BBCC can be applied to this problem to find interesting and novel patterns of customer-product purchases. In case of text data clustering, it is often the case that not all the words are discriminative of the different topics that the documents belong to. Also, the set of words is very large and the high dimensionality make the document clustering problem very challenging. BBCC can prune away non discriminative words, hence reducing the dimensionality of the data through feature selection, while simultaneously clustering the documents on the basis of a small number of discriminative words.

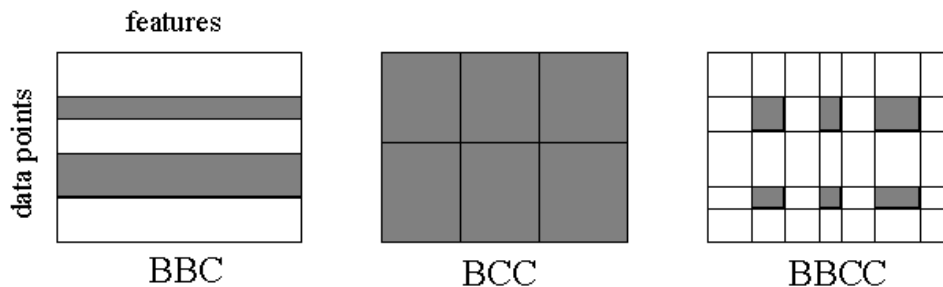


Figure 1: Nature of clusters identified by BBC, BCC, BBCC. This example has 2 row clusters and 3 column clusters (for BCC and BBCC). Shaded areas represent clustered elements, rearranged according to cluster labels, white areas denote discarded values.

Our main contributions are summarized below.

1. By performing simultaneous feature selection and “outlier removal”, BBCC provides us the ability to prune away the non-informative background of a dataset and discover dense, coherent co-clusters, i.e. Bregman co-bubbles [GG06]. This gives us better local density estimation than ordinary Bregman co-clustering (BCC).
2. BBCC inherits several key properties of the parent BCC approach: (1) applicability to 6 different schemes, (2) generalization to all Bregman divergences, including squared Euclidean distance, commonly used for clustering microarray data and I-divergence, commonly used for text data clustering [DMM03] and (3) the ability to deal with missing data values.
3. BBCC also provides a mechanism for finding a single dense co-cluster, determined on local rather than global quality, by pruning away the less informative parts of the data matrix. An exhaustive clustering algorithm like BCC is useless for this task, since it would return the entire data matrix as a single co-cluster.
4. We also extend BBCC to a more generalized co-clustering framework (Generalized BBCC), which includes a novel model selection strategy and utilizes the ability of BBCC to identify a single dense co-cluster in the dataset. Generalized BBCC can be used to find co-clusters that may be partially overlapping, automatically discover the number of co-clusters in the data and (unlike Bregman co-clustering) find variable sized, arbitrarily positioned co-clusters.

Through extensive experimental evaluation we illustrate that our proposed approach consistently gives good results. The results on the human cancer microarray datasets in Section 10.1.2 highlight the fact that our approach performs better at feature selection than even human domain knowledge based data preprocessing.

## 4 Related Work

Hartigan’s pioneering work on *direct clustering* was the first to reveal the potential of co-clustering [Har72]. In a two dimensional matrix, co-clustering aims at identifying homogeneous local patterns, each of which consists of a subset of rows and a subset of columns. In particular, co-clustering has attracted genomic researchers, because the co-clustering model is compatible with our understanding of cellular processes, where a subset of genes are coregulated under certain experimental conditions, but behave almost independently under other conditions [BDCKY03]. The paper by Madiera and Oliveira provides an extensive survey on the application of co-clustering to biological data analysis [MO04]. Cheng and Church [CC00] are considered to be the first to apply co-clustering, also called *biclustering*, to gene expression data. They proposed a greedy search heuristic that generates biclusters, one at a time, based on a homogeneity constraint, called *mean squared residue*. Since then, several similar residue based co-clustering approaches have been proposed to enhance the work of Cheng and Church. Recently, Cho et al. [CDGS04] developed two minimum sum squared co-clustering algorithms: one with its objective function based on the partitioning model proposed by Hartigan [Har72] and the other one based on the mean squared residue formulated by Cheng

and Church [CC00]. Bregman Co-clustering is a generalized co-clustering framework proposed by Banerjee et al. [BDG<sup>+</sup>07], which includes several previously developed co-clustering algorithms like information theoretic co-clustering [DMM03] and minimum sum squared co-clustering [CDGS04] as special cases.

Our approach is also related to subspace clustering, where a cluster is defined as a set of data points that are similar across a subset of the features, i.e. a subspace of the feature space. Parsons et al. present a survey of the existing subspace clustering algorithms [PHL04], which includes bottom-up grid based methods algorithms like CLIQUE [AGGR98] and iterative top-down algorithms like PROCLUS [AWY<sup>+</sup>99]. A framework for comparing subspace clusterings has been recently proposed by Patrikainen and Meila [PM06]. They formalize a set of desirable properties for such evaluation measures and propose several measures satisfying them. The proposed measures include simple and intuitive generalizations of evaluation measures for ordinary clustering such as Rand index and clustering error. These evaluation measures are applicable to the different types of subspace clusterings like axis aligned and non-axis aligned subspace clusterings, which may include overlapping clusters.

Density based algorithms use the notion of local density to define a cluster and aim at clustering only a relevant subset of the data into multiple dense clusters. DBSCAN [EKSX96] relies on the notion of density to find arbitrarily shaped clusters in large spatial databases in the presence of noise. The key idea is that for each point assigned to a cluster, the density in a specified neighborhood around the point has to exceed a certain threshold. The major limitations of DBSCAN are that it is not scalable to large, high dimensional datasets and is limited to Euclidean or related distance measures. The One Class Information Bottleneck algorithm [CC04] and the Batch Ball One Class Clustering algorithm [GG05] are efficient and scalable algorithms that can work with a large class of distance measures. However, both these algorithms find only a single dense region. The Bregman Bubble Clustering (BBC) algorithm [GG06] is a scalable, iterative relocation algorithm that simultaneously finds  $k$  dense regions in a dataset, while ignoring a specified fraction of the data points. Bubble Agglomeration is another density based clustering algorithm proposed by Barakat et. al [BJY04], which is a very simple approach to detect arbitrarily shaped clusters. The algorithm treats each data point as the center of a bubble of a fixed radius. A set of contiguous or intersecting bubbles is defined as a natural cluster or core. As the bubble radius is increased, the number of clusters decrease progressively until all the data points are assigned to a single cluster. Model selection can be performed using a reliability curve of the number of clusters verses the bubble radius, which can be used to identify the optimal bubble radius.

**Notation:** Small letters represent scalars e.g.  $a, z$ , small, bold face letters represent vectors e.g.  $\mathbf{b}, \mathbf{c}$ , capital letters like  $Z, W$  represent matrices. Individual elements of a matrix e.g.  $Z$  are represented as  $z_{ij}$  where  $i$  and  $j$  are the row and column indices respectively.

## 5 Problem Definition

Let  $m$  be the total number of rows (data points) and  $n$  the total number of columns (features). The data can be represented as an  $m \times n$  matrix  $Z$  of data points and features. Our aim is to simultaneously cluster  $s_r$  of the rows and  $s_c$  of the columns, into a grid of  $k$  row clusters and  $l$  column clusters. The co-clusters will hence be comprised of  $s_r$  rows and  $s_c$  columns and will retain  $s_r \times s_c$  of the  $m \times n$  matrix entries. Let  $\rho$  be a mapping from the  $s_r$  rows to the

$k$  row clusters and  $\gamma$  be a mapping from the  $s_c$  columns to the  $l$  column clusters. Let  $C$  be a co-clustering basis and  $d_\phi$  a suitable Bregman divergence. We want to find a co-clustering defined by  $(\rho, \gamma)$  for the basis  $C$ , and specified  $s_r$  and  $s_c$  that minimizes the following objective function

$$\sum_{g=1}^k \sum_{h=1}^l \sum_{u:\rho(u)=g} \sum_{v:\gamma(v)=h} d_\phi(z_{uv}, \hat{z}_{uv}) \quad (1)$$

where,  $z_{uv}$  is the original value in row  $u$ , column  $v$  of the matrix, assigned to row cluster  $g$  and column cluster  $h$  and  $\hat{z}_{uv}$  is the value approximated by the co-clustering solution. The objective function is the element wise error between the original and the approximated value, summed only over the clustered elements ( $s_r \times s_c$ ) of the matrix  $Z$  (additive). The reconstructed value  $\hat{z}_{uv}$  depends on the co-cluster elements, the Bregman divergence  $d_\phi$  and the basis  $C$ .

We now discuss two special cases of the above formulation, with squared Euclidean distance as the Bregman divergence. In these cases, the approximation error is given by

$$e_{ij} = (z_{ij} - \hat{z}_{ij})^2, \quad (2)$$

These two special cases of the cost function correspond exactly to the first and second residue proposed by Cho et al. [CDGS04].

- **Basis 2:** In this case, a matrix entry  $z_{ij}$  is approximated by the mean of the co-cluster it is assigned to. Let the co-cluster row and column indices be represented by sets  $I$  and  $J$  respectively. Then,  $\hat{z}_{ij} = z_{IJ}$ , where  $z_{IJ}$  is the mean of all the entries in the co-cluster i.e.,  $z_{IJ} = \frac{1}{|I||J|} \sum_{i \in I, j \in J} z_{ij}$ . The total error of all the elements within a co-cluster is zero if and only if all the elements in the co-cluster are the same or the co-cluster is trivial i.e. it has one or no entry. Basis 2 hence identifies uniform blocks of the data matrix as co-clusters.
- **Basis 6:**  $z_{ij}$  is approximated by the co-cluster row mean + the co-cluster column mean - the co-cluster mean, i.e.  $\hat{z}_{ij} = z_{iJ} + z_{IJ} - z_{IJ}$ .  $z_{iJ}$ , the mean of the entries in row  $i$  whose column indices are in  $J$  is computed as  $z_{iJ} = \frac{1}{|J|} \sum_{j \in J} z_{ij}$  and  $z_{Ij}$ , the mean of the entries in column  $j$  whose row indices are in  $I$  as  $z_{Ij} = \frac{1}{|I|} \sum_{i \in I} z_{ij}$ . The co-cluster error is zero if and only if the submatrix described by  $I$  and  $J$  is of the form  $xe^T + ey^T$  where  $e = [11 \dots 1]^T$  and both  $x$  and  $y$  are arbitrary vectors. Basis 6 can hence identify co-clusters that show a coherent trend or pattern in the data values, making it suitable for clustering gene expression data [CDGS04].

## 6 Bregman Bubble Co-clustering Algorithm

A co-clustering  $(\rho, \gamma)$ , that minimizes the objective function (1) can be obtained by an iterative algorithm similar to the Bregman co-clustering algorithm [BDG<sup>+</sup>07]. The objective function can be expressed as a sum of row or column errors, computed over the  $s_r$  assigned rows and  $s_c$  assigned columns. If row  $u$  is assigned to row cluster  $g$  i.e.  $\rho(u) = g$ , the row error is the error summed over the appropriate  $s_c$  elements in the row as

$$E_u(g) = \sum_{h=1}^l \sum_{v:\gamma(v)=h} d_\phi(z_{uv}, \hat{z}_{uv}).$$

For a fixed  $\gamma$ , the best choice of the row cluster assignment for row  $u$  is the  $g$  that minimizes this error.

$$\rho^{new}(u) = \operatorname{argmin}_g E_u(g)$$

After computing the best row cluster assignment for all the  $m$  rows, the rows are sorted in increasing order of their row errors and the top  $s_r$  rows with minimum error are selected to participate in the current row clusters. A similar approach is used to assign columns to column clusters. Note that the rows/columns that are not included in the current  $s_r/s_c$  rows/columns assigned to co-clusters are still retained since they could be included in the co-clusters in future iterations. The row cluster update step selects the fraction of rows that minimize the error among the entire set of rows. Also, the assignment of selected rows to the corresponding row clusters is done in such a way that it directly minimizes the error. Hence row cluster updates and similarly column cluster updates reduce the objective function and improve the clustering solution. Updating column cluster assignments could cause the best row assignments to change and visa versa. Optionally, the row and column cluster reassignment steps can be repeated several times, in arbitrary order until row and column cluster memberships converge.

Given the current row and column cluster assignments  $(\rho, \gamma)$ , the matrix approximation  $\hat{Z}$  has to be updated by computing the required co-cluster statistics. Updating  $\hat{Z}$  involves solving the Minimum Bregman Information (MBI) problem for the specified scheme  $C$  [BDG<sup>+</sup>07]. For example, for the special case of basis 2 and squared Euclidean distance as the Bregman divergence, the MBI solution is simply the mean of all the elements in the co-cluster. Solving the minimum Bregman information problem for the model update step is guaranteed to decrease the objective function.

The overall iterative algorithm is described in Figure 2. Step 1 minimizes the objective function due to the property of the minimum Bregman information solution, steps 2 and 3 directly minimize the objective function. The objective function hence decreases at every iteration. Since this function is bounded from below by zero, the algorithm is guaranteed to converge to a local minima.

**Weights on Data Points.** The above algorithm can be extended to handle a weight  $w_{uv}$  associated with each data point  $z_{uv}$ . The objective function in equation 1 is modified to

$$\sum_{g=1}^k \sum_{h=1}^l \sum_{u:\rho(u)=g} \sum_{v:\gamma(v)=h} w_{uv} d_\phi(z_{uv}, \hat{z}_{uv}). \quad (3)$$

The row/column cluster assignment steps and the co-cluster model update step can be easily modified to minimize the modified objective function. Incorporating weights associated with each data point into the algorithm allows it to deal with missing values. The weight for known values is set to 1 and missing values can be effectively ignored by setting their weights to 0. In general, the weight is not restricted to 0 and 1 and can take other values. This formulation allows this algorithm to deal with data uncertainties by giving comparatively lower, non-negative weights to less certain data values.

**Time Complexity.** The parent BCC algorithm for squared Euclidean distance and I-divergence is linear in the size of the input data with a time complexity of  $O(mn + mkl + nkl)$  per iteration, for all 6 bases [BDG<sup>+</sup>07]. The model update step (Step 1 in Figure 2) for squared Euclidean distance and I-divergence has a closed formed solution involving  $O(mn)$  operations. The  $mkl$  operations are due to step 2a. (Figure 2), which computes the distance of all the  $m$  rows from

**Algorithm: BBCC**Input:  $Z_{m \times n}$ ,  $s_r, s_c, k, l$ , co-clustering basis  $C$ , Bregman divergence  $d_\phi$ Output: Co-clustering  $(\rho, \gamma)$ 1. Begin with a random co-clustering  $(\rho, \gamma)$ 

2. Repeat

**Step 1**

3. Update co-cluster models

4. for  $g = 1$  to  $k$  do5. for  $h = 1$  to  $l$  do6. Update the statistics for co-cluster  $(g, h)$  based on6. the scheme  $C$  to compute new  $\hat{z}$  values

7. end for

8. end for

**Step 2**9. Update  $\rho$ 10. **2a.** Assign each row to the row cluster that minimizes the error11. for  $u = 1$  to  $m$  do12.  $\rho(u) = \operatorname{argmin}_g \sum_{h=1}^l \sum_{v:\gamma(v)=h} d_\phi(z_{uv}, \hat{z}_{uv})$ 

13. end for

14. **2b.** Choose the top  $s_r$  rows with minimum error from the  $m$  rows**Step 3**15. Update  $\gamma$ 16. **3a.** Assign each column to the column cluster that minimizes the error17. for  $v = 1$  to  $n$  do18.  $\gamma(v) = \operatorname{argmin}_h \sum_{g=1}^k \sum_{u:\rho(u)=g} d_\phi(z_{uv}, \hat{z}_{uv})$ 

19. end for

20. **3b.** Choose the top  $s_c$  columns with minimum error from the  $m$  columns

21. until convergence

22. return  $(\rho, \gamma)$ 

Figure 2: Pseudo-code for Bregman Bubble Co-clustering Meta-Algorithm

the  $k$  row clusters. This computation involves  $O(l)$  operations rather than  $O(n)$  since each row and row cluster representative can be represented in reduced form. Similarly, step 3a. involves  $O(nkl)$  operations. Since the number of row and column clusters is usually a lot lower than the number of data points and features, the row and column cluster re-assignment steps are more efficient as compared to re-assignment in one-sided clustering. BCC is hence very efficient and scalable and in practice has a lower running time than one-sided clustering [BDG<sup>+</sup>07].

The additional steps in the BBCC algorithm are the selection steps (2b. and 3b.) which choose  $s_r$  rows and  $s_c$  columns with smallest error out of all the  $m$  rows and  $n$  columns respectively. Rather than naively sorting all the row errors and selecting the top  $s_r$  rows with smallest error, which results in a time complexity of  $O(m \log m)$  for the row selection step, a more efficient approach can be used. We can take advantage of the fact that the  $s_r$  rows with smallest error out of the  $m$  rows need not themselves be sorted and can be in arbitrary

order. In this case, the problem reduces to finding the  $s_r^{th}$  largest element out of a list of  $m$  elements. Once this element is found, a single pass over the list of  $m$  elements can be performed to select the elements less than the  $s_r^{th}$  largest element. The  $s_r^{th}$  largest element out of a list of  $m$  elements can be found in linear time ( $O(m)$ ) by the efficient order statistics based algorithm proposed by Blum et al. [BFP<sup>+</sup>73]. Hence by using this approach, steps 2b. and 3b. can be performed in time  $O(m)$  and  $O(n)$  respectively. The overall complexity of BBCC is  $O(mn + mkl + nkl + m + n) = O(mn + mkl + nkl)$  per iteration. The selection steps hence do not contribute too heavily to the running time and the time complexity of BBCC is of the same order as that of BCC.

**Unification.** The BBCC framework is very general with a lot of known algorithms as its special cases. In fact, the BBC and BCC meta-algorithms are special cases of BBCC.

- BBCC with  $s_r = m$  and  $s_c = n$  is the same as BCC for all 6 schemes and a selected Bregman divergence.
- BBCC with  $l = n$ ,  $s_c = n$  and scheme 2 is the same as BBC for a selected Bregman divergence.
- BBCC with  $l = n$ ,  $s_c = n$ ,  $s_r = m$  and scheme 2 is the same as BC (Bregman Clustering) for a selected Bregman divergence.
- BBCC with  $l = n$ ,  $s_c = n$ ,  $s_r = m$ , scheme 2 and squared Euclidean distance is the same as k-means.

## 7 Bregman Bubble Co-Clustering with Pressurization

The BBCC algorithm begins with random initialization for  $\rho$  and  $\gamma$ . Random initialization could lead to poor local minima. The problem is particularly severe for small  $s_r$  and  $s_c$  since row and column clusters may not move much and the final clustering may be very heavily influenced by the initialization. This problem can be addressed by using the pressurization technique applied by BBC to overcome a similar issue [GG06]. Bregman Bubble Co-Clustering with Pressurization (BBCC-Press) begins by clustering all the data and iteratively shaving off data points and features till  $s_r$  and  $s_c$  are reached. Let  $s_r^{press} = m$  and  $s_c^{press} = n$  denote the number of data points and features to be clustered in each iteration of BBCC-Press. The pressurization technique applies BBCC iteratively with  $s_r^{press} = m$  and  $s_c^{press} = n$  in the first iteration and  $s_r^{press}$  and  $s_c^{press}$  decaying exponentially from the next iteration till  $s_r^{press} = s_r$  and  $s_c^{press} = s_c$ . The rate of decay is controlled by parameters  $\beta_{row}$  and  $\beta_{col}$  between 0 and 1. In iteration  $j$ ,

$$\begin{aligned} s_r^{press} &= s_r + \lfloor m - s_r * \beta_{row}^{j-1} \rfloor, \\ s_c^{press} &= s_c + \lfloor m - s_c * \beta_{col}^{j-1} \rfloor. \end{aligned}$$

The intuition behind pressurization is that beginning with all the data being clustered and slowly reducing the fraction of data clustered, gives the co-clusters greater mobility. Co-clusters can move around considerably from their initial positions to discover small, coherent patterns in the data. This technique hence has a better chance at improving the quality of the local minima achieved by the algorithm. For speed, each run of BBCC with different  $s_r^{press}$  and  $s_c^{press}$  need not be run to convergence and can be run for a small, fixed number of iterations. This heuristic gives competitive results in practice, without a very large increase in run time.

## 8 Generative Model for Soft BBCC

In Section 6 we described the BBCC algorithm for hard partitional cluster assignments, i.e. each row/column is assigned to either a single row/column cluster or the background. Here we focus on the more general soft co-clustering formulation, where each matrix element belongs to multiple co-clusters and to the background with different probabilities. The generative model for this formulation consists of a mixture of  $k \times l$  exponential family distributions, corresponding to the  $k \times l$  co-clusters, and a background uniform distribution, corresponding to the non-informative data matrix entries. Each element  $z_{ij}$  of the data matrix  $Z$  is assumed to be generated from this mixture model as follows

$$P(z_{ij}) = \sum_{I=1}^k \sum_{J=1}^l \alpha_{IJ} f_{\psi}(z_{ij} | \theta_{i,j,I,J}) + \alpha_0 p_0, [i]_1^m, [j]_1^n, \quad (4)$$

where  $\alpha_{IJ}$  denotes the co-cluster priors,  $f_{\psi}$  is an exponential family distribution with cumulant  $\psi(\cdot)$  and  $\theta_{i,j,I,J}$  is the natural parameter. The form of  $\theta_{i,j,I,J}$  depends on the co-clustering scheme. For scheme 2,  $\theta_{i,j,I,J} = \theta_{I,J}$  but for other schemes  $\theta_{i,j,I,J}$  also depends on  $i$  and  $j$ , e.g. for scheme 6  $\theta_{i,j,I,J} = \theta_i + \theta_j + \theta_{I,J}$  [AM07].  $\alpha_0$  and  $p_0$  denote the prior probability and the probability density of the uniform distribution. By assuming that the matrix elements are generated i.i.d with weights  $w_{ij}$ , the incomplete data log-likelihood is given by

$$L(\Theta | Z) = \sum_{i=1}^m \sum_{j=1}^n w_{ij} \log P(z_{ij}), \quad (5)$$

where  $\Theta$  denotes all the model parameters. It is however not possible to directly maximize this data log-likelihood and we associate latent variables  $\rho(i)$  and  $\gamma(j)$ , denoting cluster membership, with each element  $z_{ij}$ .  $\rho(i)$  and  $\gamma(j)$  take values from 0 to  $k$  and 0 to  $l$  respectively, where  $\rho(i) = 0, \gamma(j) = 0$  are used to denote membership to the uniform background.

We use the standard EM technique to fit the mixture model described above. Similar to the analysis by Gupta and Ghosh [GG06], we assume that  $p_0$  is set to an appropriately selected fixed value. Hence, the parameters we optimize over are the priors and the parameters of the exponential distributions. We first construct the free energy function [NH98] as a sum of the expected complete data log-likelihood and the entropy of the latent variables with respect to a distribution  $\tilde{p}(\rho(i), \gamma(j))$

$$F(\tilde{p}, \Theta) = \sum_{ij} w_{ij} E_{\tilde{p}_{ij}} \log P(z_{ij}, \rho(i), \gamma(j)) + \sum_{ij} w_{ij} H(\tilde{p}_{ij}), \text{ where} \quad (6)$$

$$E_{\tilde{p}_{ij}} \log P(z_{ij}, \rho(i), \gamma(j)) = \sum_{IJ} \tilde{p}_{ij}(I, J) \log(\alpha_{IJ} f_{\psi}(z_{ij} | \theta_{i,j,I,J})) + \tilde{p}_{ij}(0, 0) \log \alpha_0 p_0,$$

$$H(\tilde{p}_{ij}) = \sum_{IJ} \tilde{p}_{ij}(I, J) \log \tilde{p}_{ij}(I, J) + \tilde{p}_{ij}(0, 0) \log \tilde{p}_{ij}(0, 0).$$

It can be shown that maximizing  $F(\tilde{p}, \Theta)$  with respect to  $\tilde{p}$  and  $\Theta$  is equivalent to maximizing the data log-likelihood given by Equation 5 [NH98].  $F(\tilde{p}, \Theta)$  can be maximized by the EM procedure, which alternates between maximizing  $F$  w.r.t.  $\tilde{p}$  for fixed  $\Theta$  (E-step) and maximizing

$F$  w.r.t.  $\Theta$  for fixed  $\tilde{p}$  (M-step) and is guaranteed to converge to a local maximum of  $F$ . The details of the E and M step are as follows.

**E-step:** Here we have the constraint  $\sum_{IJ} \tilde{p}_{ij}(I, J) + \tilde{p}_{ij}(0, 0) = 1, \forall i, j$ . By introducing Lagrange multipliers for the equality constraints and differentiating  $F$  w.r.t.  $\tilde{p}$  one can arrive at the update step given below.

$$\tilde{p}_{ij}(I, J) = \frac{\alpha_{IJ} f_{\psi}(z_{ij}; \theta_{i,i,I,J})}{\sum_{IJ} \alpha_{IJ} f_{\psi}(z_{ij}; \theta_{i,i,I,J}) + \alpha_0 p_0}, [I]_1^k, [J]_1^l, \quad (7)$$

$$\tilde{p}_{ij}(0, 0) = \frac{\alpha_0 p_0}{\sum_{IJ} \alpha_{IJ} f_{\psi}(z_{ij}; \theta_{i,i,I,J}) + \alpha_0 p_0}.$$

**M-step:**

Updating priors: The constraint on the priors of the mixture components is  $\sum_{IJ} \alpha_{IJ} + \alpha_0 = 1$ . It can be shown that the update step is as follows

$$\alpha_{IJ} = \frac{\sum_{ij} w_{ij} \tilde{p}_{ij}(I, J)}{\sum_{ij} w_{ij}}, [I]_1^k, [J]_1^l,$$

$$\alpha_0 = \frac{\sum_{ij} w_{ij} \tilde{p}_{ij}(0, 0)}{\sum_{ij} w_{ij}}.$$

Updating parameters: The mixture component parameters can be estimated by differentiating  $F$  w.r.t.  $\theta_{i,j,I,J}$  and equating to 0 as follows

$$\sum_{ij} w_{ij} \sum_{IJ} \tilde{p}_{ij}(I, J) \nabla_{\theta_{i,j,I,J}} \log(\alpha_{IJ} f_{\psi}(z_{ij}; \theta_{i,i,I,J})) = 0.$$

We now consider the special case of a mixture of Gaussian components with fixed variance  $\sigma^2$  and co-clustering basis 2. According to the bijection theorem between exponential families and Bregman divergences [BDG<sup>+</sup>07], the Gaussian components correspond to using squared Euclidean distance as the Bregman divergence. Since we are using basis 2, the natural parameter  $\theta_{i,j,I,J} = \theta_{I,J}$  and  $f_{\psi}(z_{ij} | \theta_{i,j,I,J}) = \frac{1}{\sqrt{(2\pi\sigma^2)}} e^{-\frac{(z_{ij} - \theta_{IJ})^2}{2\sigma^2}}$ . The update step for  $\theta_{IJ}$  can then be derived as

$$\theta_{IJ} = \frac{\sum_{ij} w_{ij} \tilde{p}_{ij}(I, J) z_{ij}}{\sum_{ij} w_{ij} \tilde{p}_{ij}(I, J)}.$$

One can observe that the E and M update steps illustrated above are generalizations of the update steps for the soft BBC algorithm that deals with the one-sided clustering formulation [GG06]. The soft BBCC model described above is very flexible since it does not impose any structure on the co-clusters. However, the standard EM algorithm used to estimate the model is not scalable and can be very slow in practice. To address this issue, one can impose structural constraints on the generative mixture model. We do so by restricting  $\tilde{p}_{ij}$  to the form  $\tilde{p}_{ij}(\rho(i), \gamma(j)) = \tilde{p}_i(\rho(i)) \tilde{p}_j(\gamma(j))$ , i.e.  $\rho(i)$  and  $\gamma(j)$  are independent. Hence, we assume that every row  $i$  of the data matrix  $Z$  belongs to row clusters  $I = 1$  to  $k$  and the background row cluster  $I = 0$  with probabilities  $\tilde{p}_i(I)$  and similarly every column  $j$  belongs to column clusters  $J = 1$  to  $l$  and the background column cluster  $J = 0$  with probabilities  $\tilde{p}_j(J)$ . By decoupling the row and column cluster assignments, we can now estimate the model very efficiently by an

EM based algorithm. The steps of the algorithm are illustrated in Figure 3. Each step monotonically decreases the free energy function, finally converging to a locally optimal solution. This algorithm is a generalization of the hard BBCC algorithm described in Section 6 for soft cluster assignments.

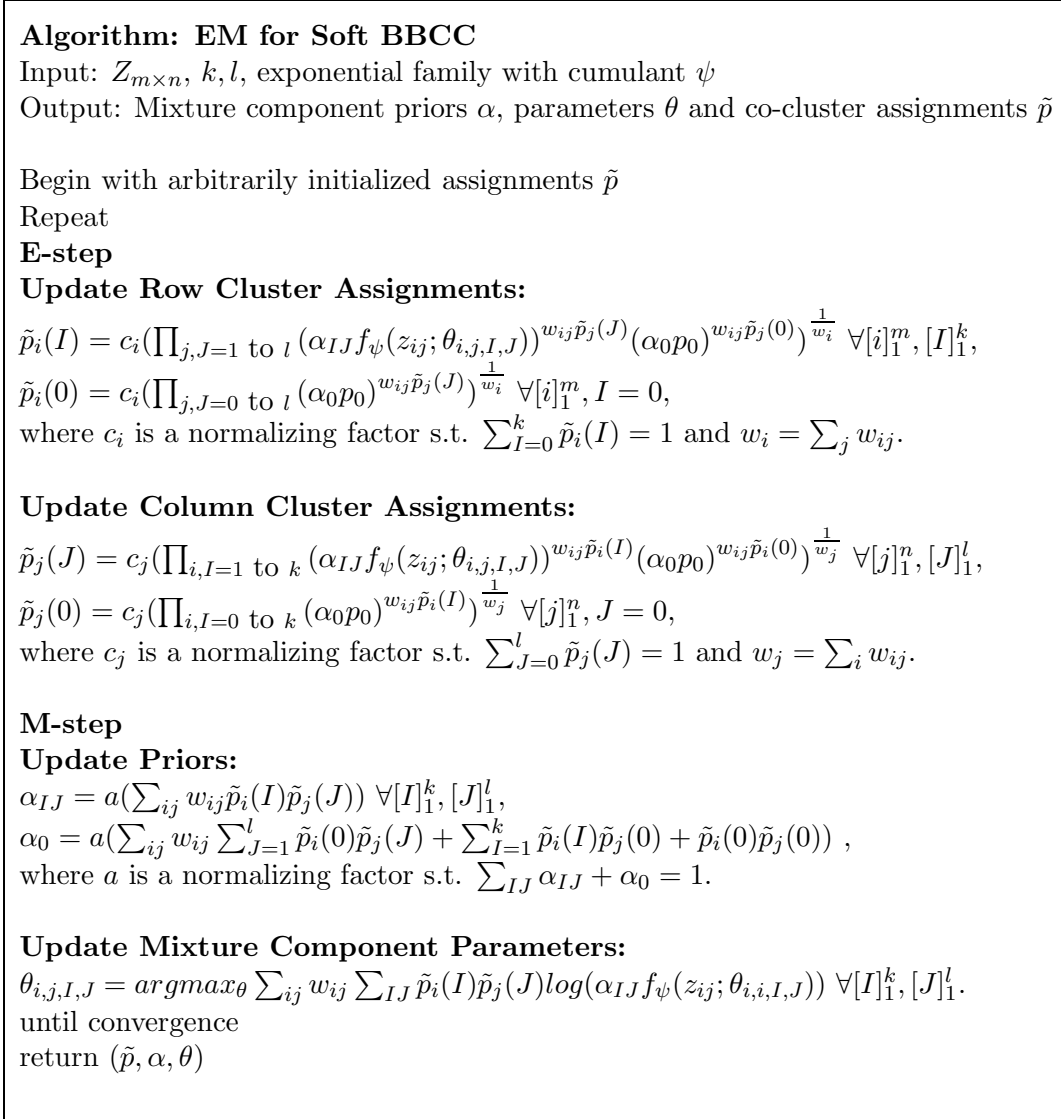


Figure 3: EM algorithm for estimation of soft BBCC model

## 9 Experimental Evaluation on Synthetic Data

The BBCC and BBCC-Press algorithms were first evaluated on a number of synthetic datasets. These initial experiments were carried out to validate the BBCC approach when the generative model of the data matches the model assumptions and to determine the benefit obtained by BBCC over BBC and BCC in a controlled setting. The synthetic data was generated by the following procedure:

1. A matrix  $Z$  of values randomly selected from a uniform distribution (range 0-10) forms the background.
2. Coherent blocks of values representing co-clusters are embedded in  $Z$ , to form a set of co-clusters involving a small fraction of the data, arranged in a grid structure as illustrated in Figure 1. The blocks are generated specific to the co-clustering basis. We include experiments with bases 2 and 6, which have been used previously for co-clustering of microarray data [CDGS04], [CC00]. The specific data generation process for the two bases is described below.

**Basis 2.** A block for a basis 2 co-cluster is generated by randomly selecting the co-cluster values from a Gaussian distribution with a mean in the range 0-10. Different co-cluster blocks are generated from different Gaussian distributions. A co-cluster is hence a uniform block of constant values with additive Gaussian noise. The elements in the block are then spatially rearranged such that the values close to the center of the block are the closest to the block mean and the deviation of the values from the mean increases outwards from the center of the block. This is to ensure that the data points (rows) that are spatially close to the center of the cluster are more similar, in a certain subspace of the feature space, to the cluster centroid than points further away from the cluster center. Similarly, the features (columns) that are spatially close to the cluster center are very similar and coherent. For small  $s_r$  and  $s_c$  we expect the BBCC algorithm to discover the coherent set of data points and features, close to the co-cluster center, i.e. the roughly uniform interior of the co-cluster.

**Basis 6.** Basis 6 can identify co-clusters that show a coherent trend or pattern in the data values. To generate a block of values for a basis 6 co-cluster of size  $r$  by  $c$ , we first generate a vector (1 by  $c$ ) of random values in the range 0-10. Each of the  $r$  rows of the block are generated by shifting this vector by a randomly selected constant factor. All the rows in the co-cluster hence have the same pattern across all the columns in the co-cluster. The values in the block are then perturbed by adding Gaussian noise.

3. Finally, the rows and columns of  $Z$  are randomly permuted.

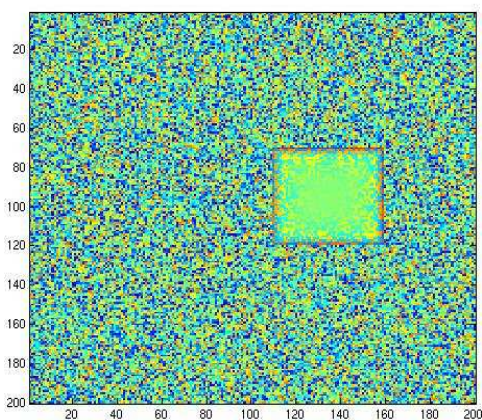
Table 1 describes the synthetic datasets that were used for experimentation. Datasets 1 and 4 have only one co-cluster, that is they have one coherent block embedded in a data matrix representing background noise. The plots of the data matrices for datasets 1-3 are illustrated in Figure 4.

### 9.1 Results on Synthetic Data

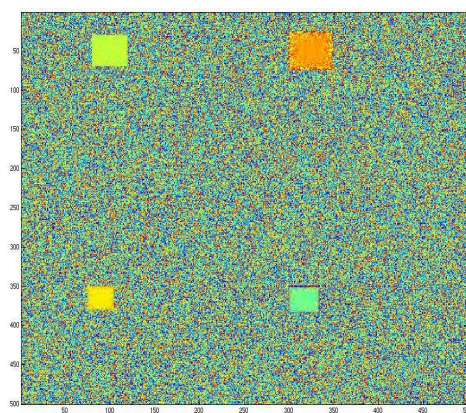
Here we compare the performance of the BBCC algorithm with the baseline, one-sided Bregmen clustering (BC), Bregman co-clustering (BCC), and Bregman Bubble clustering with pressur-

Data	$m,n$	$k,l$	sizes of co-clusters	scheme
Dataset 1	200, 200	1,1	50 by 50	2
Dataset 2	500, 500	2,2	40 by 40, 50 by 50, 30 by 30, 35 by 35	2
Dataset 3	200, 500	2,2	50 by 40, 70 by 50, 30 by 30, 35 by 45	6
Dataset 4	200, 200	1,1	50 by 50	6
Dataset 5	200, 200	2,2	50 by 40, 70 by 50, 30 by 30, 35 by 45	6

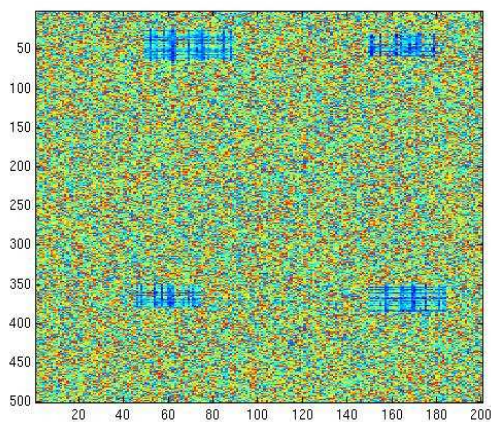
Table 1: Synthetic Datasets



(a) Dataset 1



(b) Dataset 2



(c) Dataset 3

Figure 4: Plots of Synthetic Datasets (before permuting randomly)

ization (BBC-Press) [GG06]. BC and BCC assign all the rows to row clusters, so to compare these algorithms with BBCC a post processing step is required. After convergence, this step ranks the rows in terms of their distance to the corresponding row cluster representatives and chooses the required fraction of rows as the top fraction from the ranked list. A similar post processing step can be performed to compare the column clustering quality when only a fraction

of the columns are clustered.

**Clustering evaluation measure.** We evaluate the quality of the row/column clustering solutions using the average cluster purity measure. To compute purity, each cluster is assigned to the class which is most frequent in the cluster. The average cluster purity is then the sum of correctly assigned points across all clusters, divided by the total number of points. The average cluster purity for row clusters is defined as follows.

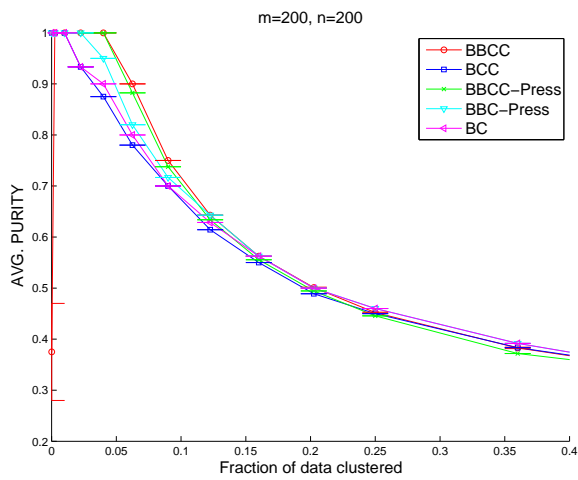
$$\text{Avg. cluster purity} = \frac{1}{m} \left( \sum_{i=1}^k t_{ij} \right),$$

where  $m$  is the total number of rows,  $k$  is the number of row clusters, and  $t_{ij}$  denotes the number of rows of cluster  $i$  correctly assigned to class  $j$ , where class  $j$  is such that  $t_{ij}$  is maximum among all  $j$ .

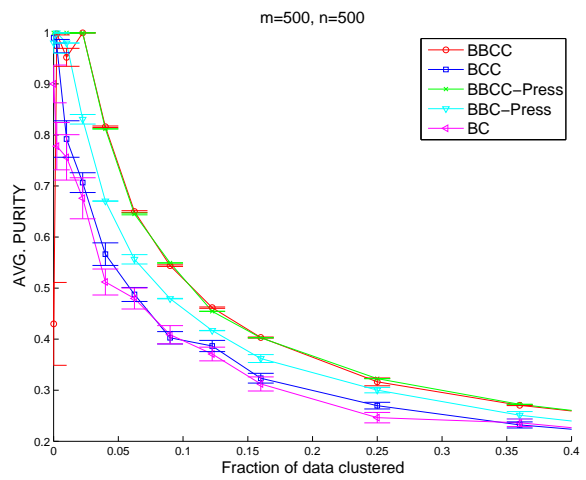
**BBCC - Scheme 2.** Figure 5 compares BBCC (scheme 2), BBCC-Press (scheme 2), BCC (scheme 2), BBC-Press and BC on synthetic datasets 1 and 2, with squared Euclidean distance as the Bregman divergence. The choice of scheme 2 and squared Euclidean distance closely match the data generation process. The number of rows and columns clustered ( $s_r$  and  $s_c$ ) are varied from  $m$  and  $n$  to very small values. The X-axis represents the fraction  $x$  of the values in the data matrix that are clustered, where  $x = \frac{s_r s_c}{mn}$ . If  $s_r^{true}$  and  $s_c^{true}$  are the actual number of rows and columns assigned to clusters according to the ground truth, then  $s_r$  and  $s_c$  are proportionally reduced from  $m$  and  $n$  to  $s_r^{true}$  and  $s_c^{true}$  as follows:  $s_r$  at the  $i^{\text{th}}$  point of the graph is given by  $s_r^i = m - (I - i) \frac{m - s_r^{true}}{I}$  and similarly  $s_c^i = n - (I - i) \frac{n - s_c^{true}}{I}$ , where  $I$  is the number of intervals. At the value 1 on the X-axis all the rows and columns are clustered and BBCC is the same as BCC. The Y-axis measures the average cluster purity of the row clusters. Both these datasets are easy and have small co-clusters, hence all the algorithms do well at small fractions of the data clustered. Overall, BBCC and BBCC-Press do slightly better than the other approaches. On these datasets, we also compared the cluster purity of the column clusters for different fractions of data clustered and found similar trends in the performance of the different approaches.

**BBCC - Scheme 6.** Figure 6 compares BBCC (scheme 6), BBCC-Press (scheme 6), BCC (scheme 6), BBC-Press and BC on synthetic datasets 3,4 and 5, with squared Euclidean distance. These datasets have blocks generated according to scheme 6 and hence the generative model of the data matches the algorithm model assumptions for BBCC and BCC. BBCC-Press has the highest row and column cluster accuracy on dataset 3 and does significantly better than the other approaches. BBCC-Press and BBCC both do better than the others on datasets 4 and 5 as well.

**BBCC - Scheme 2 with scheme 6 data.** Figure 7 displays the results of the co-clustering algorithms with scheme 2 on synthetic datasets 3,4 and 5, in order to compare the performance of the algorithms when the generative model does not match the model assumptions, which could be very likely in the case of real data. In this case the performance of the co-clustering approaches drops on all the datasets, as compared to the results in Figure 6. On dataset 3, BBCC-Press does significantly better than the other approaches, while on datasets 4 and 5 BBC-Press does slightly better than BBCC-Press and BBCC.

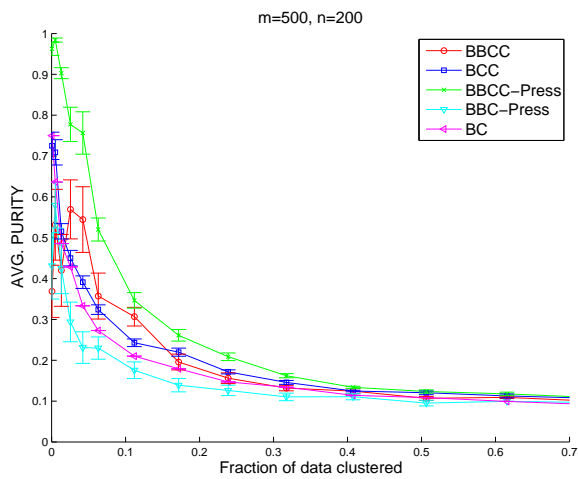


(a) Dataset 1 (Accuracy of rows)

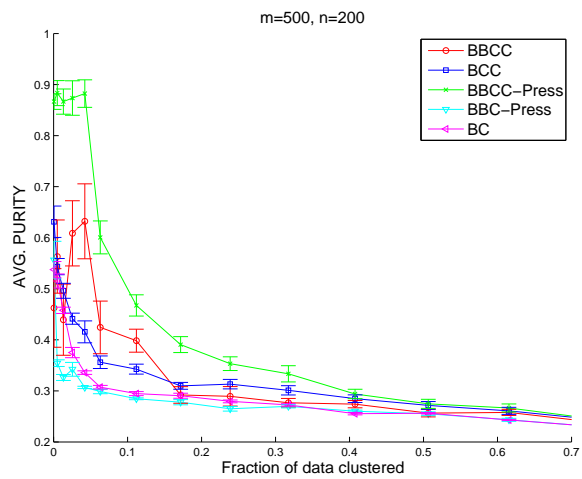


(b) Dataset 2 (Accuracy of rows)

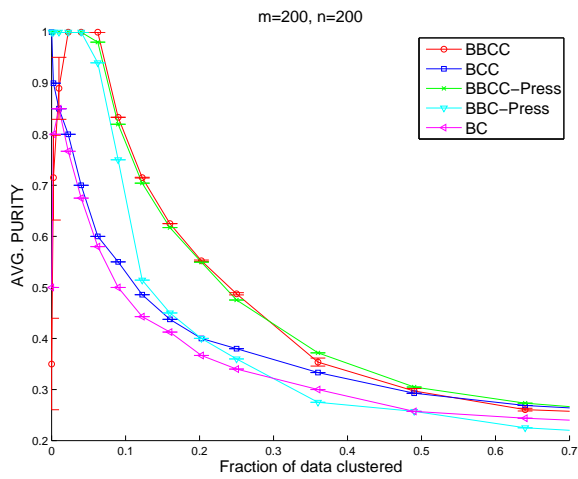
Figure 5: Synthetic data with scheme 2, using co-clustering scheme 2 with squared Euclidean distance.



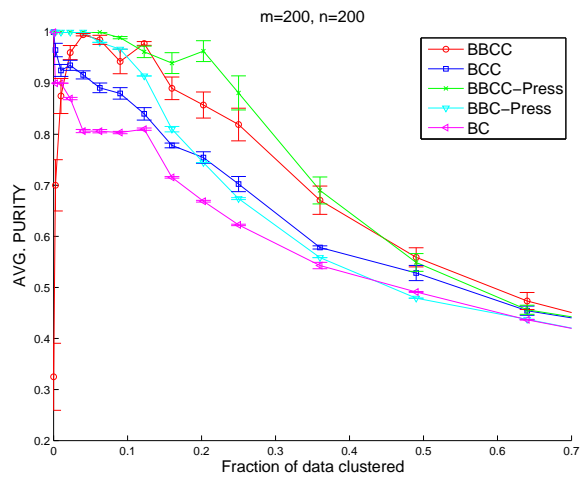
(a) Dataset 3 (Accuracy of rows)



(b) Dataset 3 (Accuracy of columns)

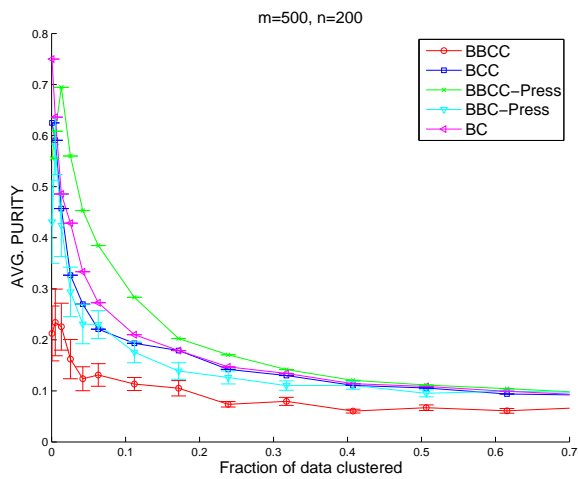


(c) Dataset 4 (Accuracy of rows)

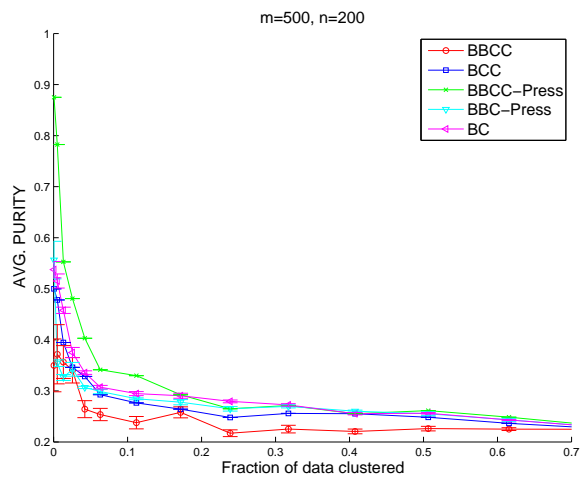


(d) Dataset 5 (Accuracy of rows)

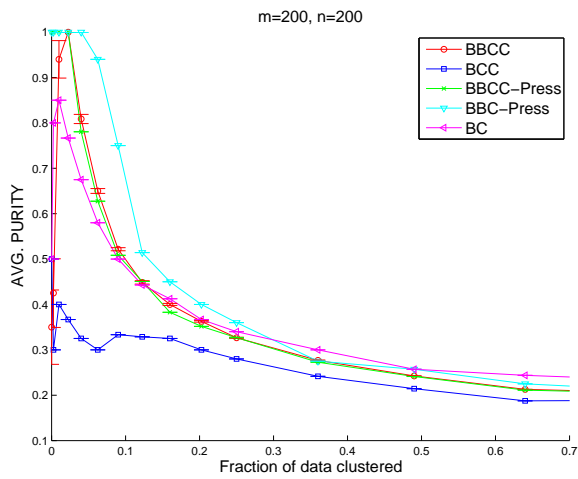
Figure 6: Synthetic data with scheme 6, using co-clustering scheme 6 with squared Euclidean distance.



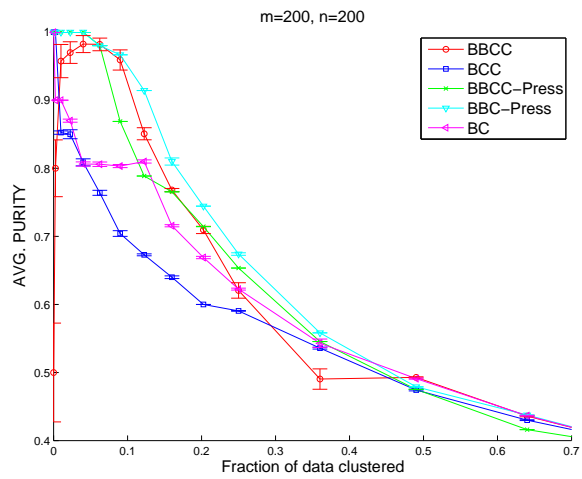
(a) Dataset 3 (Accuracy of rows)



(b) Dataset 3 (Accuracy of columns)



(c) Dataset 4 (Accuracy of rows)



(d) Dataset 5 (Accuracy of rows)

Figure 7: Synthetic data with scheme 6, using co-clustering scheme 2 with squared Euclidean distance.

## 10 Experimental Results on Real Microarray Datasets

BBCC is applicable to several real problems like text data clustering and market basket analysis as described in Section 2. Here we focus on one example; the application of BBCC to clustering microarray data. Microarray data is represented as a matrix with rows representing individual genes and columns representing samples or experiments. In this section we evaluate the performance of BBCC in two different applications related to clustering of microarray data, (i) feature selection of genes (rows) in order to cluster samples (columns), which is evaluated on human cancer datasets in Section 10.1 and (ii) pruning irrelevant genes and experiments while clustering the relevant genes, which is discussed in Section 10.2 and tested on the Lee dataset.

### 10.1 Human Cancer Gene Expression Data

In this section, we apply the proposed BBCC approach to four publicly available human cancer microarray datasets: Colon cancer [ABN<sup>+</sup>99], Leukemia [GST<sup>+</sup>99], Lung cancer [ASS<sup>+</sup>02], and MLL [ASS<sup>+</sup>02], all generated using Affymetrix technology.

**Colon Cancer.** The Colon Cancer dataset was created by Alon et al. [ABN<sup>+</sup>99] by selecting 2000 genes with highest minimal intensity across the samples from a total of 6500 human genes. The 62 samples consist of tumorous (40 samples) and normal (22 samples) colon tissues. The gene expression values were transformed by taking the base-10 logarithm.

**Leukemia.** The Leukemia dataset by Golub et al. [GST<sup>+</sup>99] consists of the expression levels of 7129 genes from 72 samples diagnosed with leukemia. Each sample belongs to either acute lymphoblastic leukemia (ALL, 47 samples) or acute myeloid leukemia (AML, 25 samples).

**Lung cancer.** Gordon et al. [GJH<sup>+</sup>02] provide this dataset with 12533 genes and 181 human tissue samples consisting of malignant pleural mesothelioma (MPM, 31 samples) and adenocarcinoma (ADCA, 150 samples with 139 patient tumors and 11 duplicates) of the lung.

**MLL dataset.** Armstrong et al. [ASS<sup>+</sup>02] obtained gene expression values of 12582 genes over 72 samples. Each sample was diagnosed by pathologists as one of the three types of leukemia: acute lymphoblastic leukemia (ALL, 24 samples), acute myeloid leukemia (AML, 28 samples), and mixed-lineage leukemia (MLL, 20 samples).

Table 2 summarizes information about the datasets, including the number of samples, genes and classes. True class labels are available for the samples (columns), however no ground truth is available for the genes (rows). In this application, the aim is to cluster all the samples, using the expression values of the genes as features. These publicly available gene expression matrices have already been preprocessed in various ways using image analysis, expression quantization, normalization, and screening out. Therefore, the numerical values in each dataset are very different, the number of genes are in thousands, while the number of samples are relatively small, often less than a hundred. Moreover, many of the genes are non-informative and redundant. This makes feature selection an important issue.

The objective of applying BBCC to this data is to co-cluster the genes and the samples and perform automatic feature selection along the “gene” axis. We expect that clustering and feature selection of the genes will bring about a substantial boost in the quality of the sample clusters. Additionally, the co-clusters provide explanatory power by describing the gene patterns that distinguish between the sample clusters, something that is not directly possible with regular one-sided clustering of the samples. In this application, BBCC is run in such a

way that all the samples are clustered and pruning is carried out only on the genes. The hope is that BBCC will identify the most discriminative genes while simultaneously co-clustering the data, and ignoring the less useful genes. Eliminating non-informative and redundant genes is expected to improve the accuracy of the sample clusters with respect to their true class labels. Note that feature selection interleaved with the clustering in this manner is completely automatic and data driven as compared to the use of domain knowledge to select the best feature set as a preprocessing step. We compare the sample cluster accuracy of BBCC with BCC, which uses all the genes to obtain a co-clustering of genes and samples, as well as one sided clustering (BC), which uses all the genes as features to cluster the samples. Along with an accurate clustering of the samples, BBCC also generates coherent gene clusters. A qualitative evaluation of the gene clusters is presented in Section 10.1.4.

**Data Transformation.** Raw data values have a limitation that they do not disclose how they vary from the central tendency of the distribution. Therefore, the transformation of raw data is considered to be one of the most important data preprocessing steps since the variance of a variable will determine its importance in a given model [SLM94]. An early study on preprocessing by Harshman and Lundy emphasizes the importance of data transformation [HL84]. Kluger et al. experiment with different normalizations of genes and conditions with the objective of discarding the irrelevant constant background noise in microarray data [KBCG03]. Cho et al. formally analyze the effect of various data transformations on their two co-clustering residue measures [CD07], where they show that through column standardization (CS), the second residue is able to capture both scaling and shifting patterns. Therefore, in our experiments, we use CS as a data preprocessing step. Column standardization is defined as

$$a'_{ij} = \frac{a_{ij} - \mu_j}{\sigma_j}$$

for  $i = 1, \dots, m$  and  $j = 1, \dots, n$ , where  $\mu_j = \frac{1}{m} \sum_{i=1}^m a_{ij}$  and  $\sigma_j^2 = \frac{1}{m} \sum_{i=1}^m (a_{ij} - \mu_j)^2$ . Column standardization results in each column having zero mean and unit variance, and is also called “autoscaling”. Through this standardization, the relative variation in intensity is emphasized. Since CS is a linear transformation, the relative positions of observations and the shape of the original distribution is retained.

**Sample clustering evaluation measure.** To evaluate the performance of the sample clustering solutions, we quantify cluster quality using the accuracy of the cluster labels with respect to the true class labels. Since we know the actual number of sample classes, we set the number of sample clusters equal to the number of classes in all our experiments. The cluster accuracy for the sample clusters is defined as follows.

$$Accuracy(\%) = \frac{1}{n} \left( \sum_{i=1}^l t_i \right) \times 100,$$

where  $n$  is the total number of samples,  $l$  is the number of sample clusters, and  $t_i$  denotes the numbers of the samples correctly clustered into a sample class  $i$ . We first form a confusion matrix where  $(i, j)^{\text{th}}$  entry is the number of samples in cluster  $i$  that belong to the true class  $j$ . Each  $t_i$  is a diagonal element of the corresponding confusion matrix whose cluster labels are permuted so that the sum of diagonal elements is maximized.

Table 2: Description of the human cancer microarray datasets.

Dataset	Colon	Leukemia	Lung	MLL
No. of genes	2000	7129	12533	12582
No. of samples	62	72	181	72
No. of sample classes	2	2	2	3
Sample class names	Normal(20) Tumor(42)	ALL(47) AML(25)	ADCA(150) MPM(31)	ALL(24) AML(25) MLL(23)

### 10.1.1 Results on Human Cancer Datasets

We compare the accuracy of the sample clusters obtained by BBCC with BCC and one-sided clustering (BC) on the cancer datasets described in Table 2. Note that BC here is k-means clustering on the samples, with all the genes as features. Figures 8, 9, 10 and 11 illustrate the sample cluster accuracy of the algorithms at different fractions of the genes clustered, averaged over 20 trials. The displayed results are on the column standardized datasets, since we obtained better accuracy on all the datasets with column standardization than without. Both BCC and BBCC use scheme 2 with squared Euclidean distance on the Leukemia, Lung and MLL datasets. Scheme 2 is a better choice in this case as compared to scheme 6 since these datasets contain a very large number of genes, many of which are known to be irrelevant and noisy and scheme 6 might overfit the noise in the data. Our experiments support this hypothesis since we found scheme 2 consistently outperforming scheme 6 on these datasets. On the Colon dataset BCC and BBCC perform better with scheme 6 since this dataset includes a relatively smaller, more carefully selected set of genes.  $k$  is set to 100 for BCC and BBCC-Press.

On all these datasets, BBCC-Press performs much better than BC. On Leukemia and Colon, BBCC-Press does a lot better than BCC as well (Figures 8, 9). Since these datasets are known to show coherent patterns involving only a small fraction of all the genes, the automatic feature selection by BBCC-Press contributes to its good performance. BC and BCC perform poorly due to the high redundancy and noise levels of the genes. BBCC-Press does much better as it can automatically prune away non-discriminative and noisy genes.

### 10.1.2 Results on Reduced (Filtered) Cancer Datasets

The human cancer datasets described in Table 2 have been analyzed by several researchers, who have addressed the problem of noisy and redundant genes by simple preprocessing heuristics, based on domain knowledge commonly adopted in microarray experiments [BJ02, DB02, DF02]. These researchers used differential expression to filter redundant genes, and have made available reduced versions of these datasets containing only the most discriminative genes. We evaluated BBCC-Press on these reduced datasets to determine whether the sets of reduced genes still contain non-informative genes, that can be pruned further, leading to a larger improvement in sample cluster accuracy. The reduced datasets were generated by filtering genes whose relative deviation ( $|max/min|$ ) or absolute deviation ( $|max - min|$ ) were less than predefined values, where  $max$  and  $min$  refer respectively to the maximum and minimum expression levels for a particular gene across all samples. The following describes the specific preprocessing steps

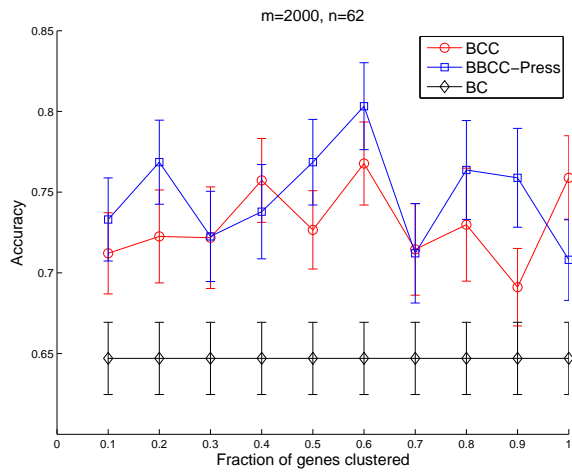


Figure 8: Whole Colon data

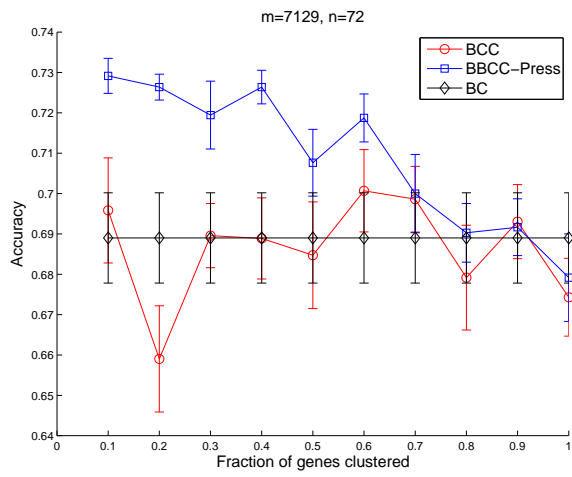


Figure 9: Whole Leukemia data

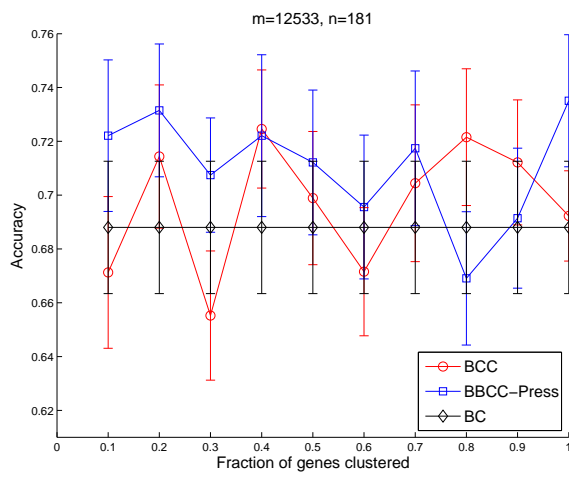


Figure 10: Whole Lung Cancer data

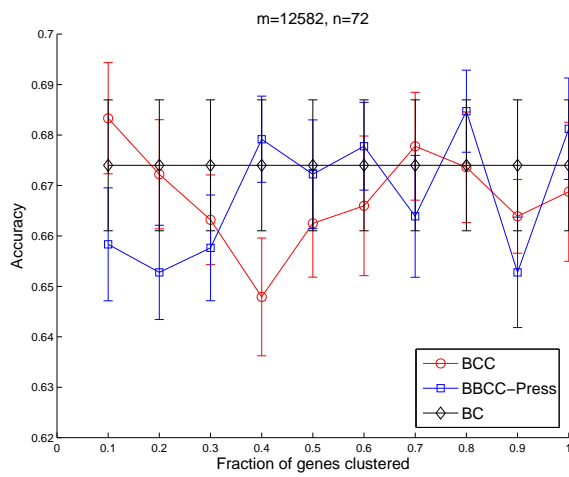


Figure 11: Whole MLL data

taken to generate the reduced version of each human cancer dataset. Table 3 summarizes the filtering parameter values and the number of genes left in the reduced datasets.

**Colon Cancer.** Genes with  $|max/min| < 15$  and  $|max - min| < 500$  were removed, leaving behind a reduced set of 1096 genes.

**Leukemia.** The reduced Leukemia dataset consists of 3571 genes, obtained by thresholding genes with a floor of 100 and ceiling of 16000 and then further screening out genes with  $|max/min| < 5$  and  $|max - min| < 500$ .

**Lung cancer.** The reduced Lung Cancer dataset has a total of 2401 genes with  $|max - min| \geq 600$ , obtained by screening out genes by fixing the relative deviation cut-off to 5 and varying the absolute deviation value so that the genes are pruned to 1/5-th.

**MLL dataset.** The reduced MLL dataset was generated using similar preprocessing steps as for the reduced Lung Cancer dataset and consists of 2474 genes with  $|max - min| \geq 5500$ .

Table 3: Description of reduced human cancer microarray datasets

Dataset	Colon	Leukemia	Lung	MLL
No. of original genes	2000	7129	12533	12582
Relative deviation threshold, $ max/min $	15	5	5	5
Absolute deviation threshold, $ max - min $	500	500	600	5500
No. of remaining genes	1096	3571	2401	2474

We compare the accuracy of the sample clusters obtained by BBCC with BCC and BC on the reduced cancer datasets described in Table 3. Figures 12, 13, 14 and 15 illustrate the sample cluster accuracy of the algorithms at different fractions of the genes clustered. In order to evaluate the effect of column standardization, we additionally compare the algorithms on the raw datasets, with no data transformations. The first figure in each set displays the clustering results with no data transformation (NT), whereas the second figure displays results on the column standardized (CS) dataset. Here BCC and BBCC use scheme 6 with squared Euclidean distance and  $k = 20$ . The results are averaged over 20 runs. Overall the accuracy of all the algorithms improves since the datasets used here are preprocessed based on domain knowledge and include a more careful selection of genes. On all the datasets BBCC-Press does better than BCC, indicating that the reduced set of genes still contains some less informative genes, which when pruned improves performance further. The results are most dramatic on the Lung Cancer dataset (Figure 14), where BBCC-Press is significantly better than BCC and one sided clustering (BC). On the other datasets, BC does not perform much worse than BBCC-Press, which is not surprising since these datasets include genes carefully selected using domain knowledge.

### 10.1.3 Visual Analysis of Co-clusters

In this section we visually evaluate the co-clusters discovered by BCC and BBCC-Press by suitably plotting the matrix of data values. The genes are sorted in increasing order of their distance to their corresponding row cluster centroids and the top 100 genes are selected for visualization. Only the top 100 genes are selected, since a larger number of genes reduces the resolution of the plot, making it difficult to observe coherent co-cluster patterns. The plot

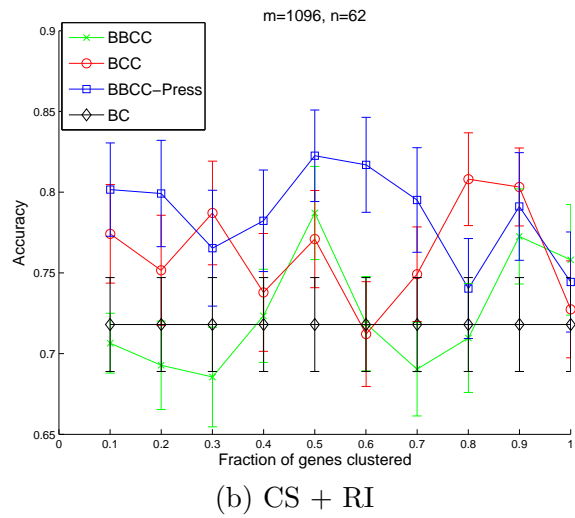
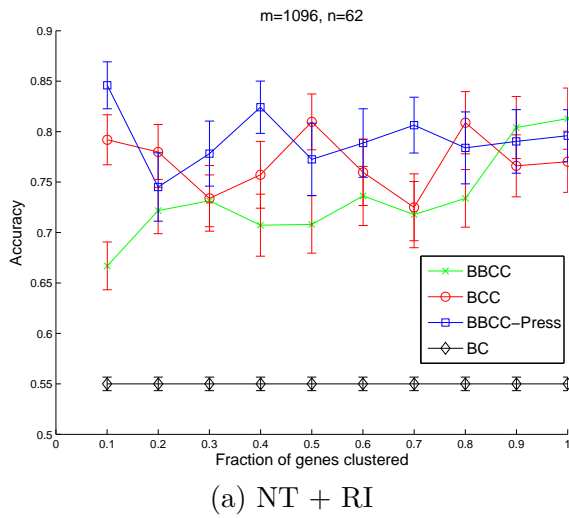


Figure 12: Colon data

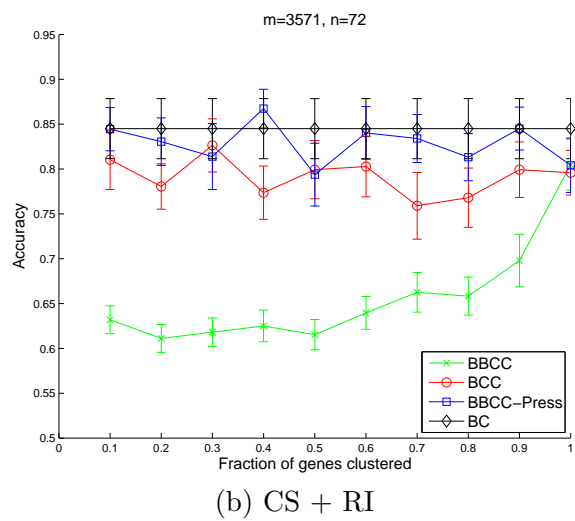
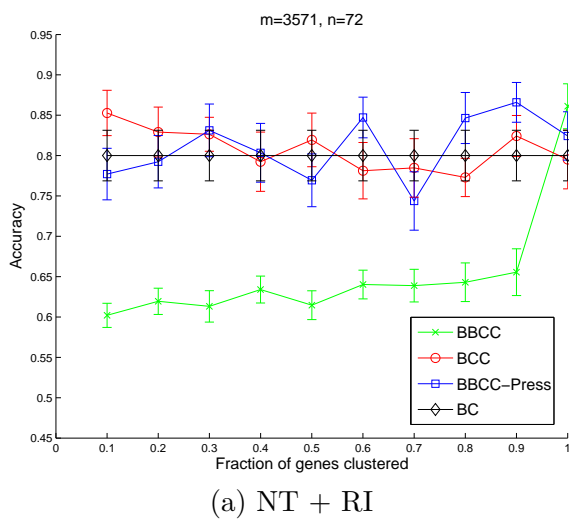
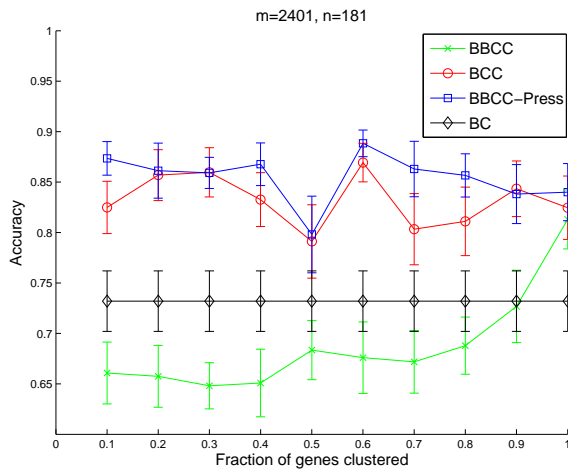
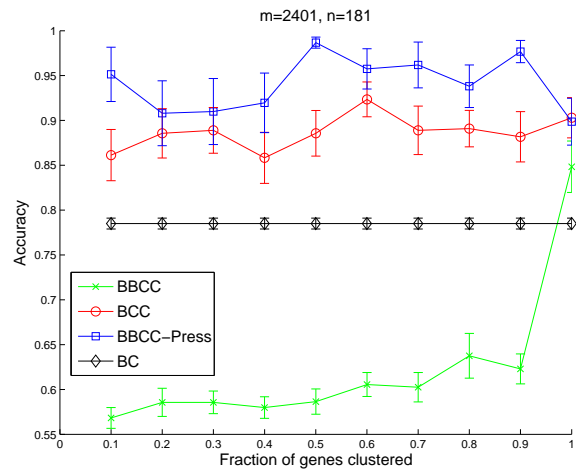


Figure 13: Leukemia data

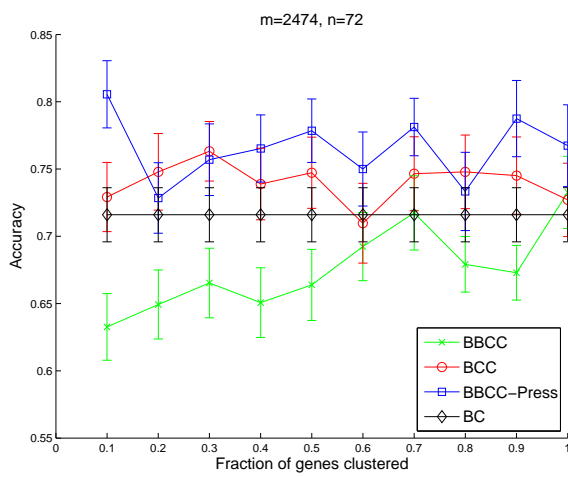


(a) NT + RI

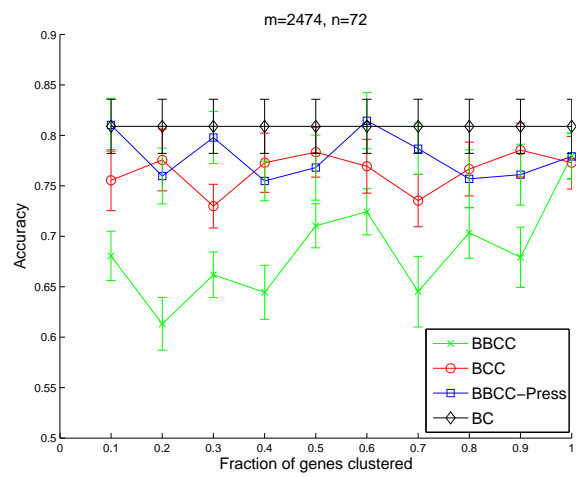


(b) CS + RI

Figure 14: Lung Cancer data



(a) NT + RI



(b) CS + RI

Figure 15: MLL data

includes the selected genes (rows) and all the samples (columns) of the data matrix, rearranged according to their row and column cluster labels. We now display a few such interesting plots.

The following figures display the plots of the rearranged data matrices obtained by BCC and BBCC-Press after clustering 10% of the genes and all the samples of the column standardized, reduced human cancer datasets. On the MLL dataset, the plot produced by BBCC-Press (Figure 16(b)) shows 3 distinct sample clusters, visually identifiable based on the expression patterns of only 100 genes. All the co-clusters appear to be very coherent and consist of highly correlated expression patterns. In contrast, in the BCC plot (Figure 16(a)) a very clear distinction of the 3 sample clusters is not visible and particularly, the first row cluster does not seem to be very coherent or discriminative with respect to the sample clusters. Lung cancer is a very imbalanced dataset with 31 samples belonging to one cluster and the remaining 150 belonging to the other. Figure 17 indicates that the expression patterns of the selected 100 genes identify the 2 sample clusters more accurately with BBCC-Press than BCC. In the BBCC-Press plot (Figure 17(b)) one can observe that the second sample cluster contains exactly 31 samples distinguished by a highly correlated gene expression pattern. On the Leukemia dataset (Figure 18), one can see the difference in the nature of the clusters identified by BBCC-Press and BCC. While BCC identifies large co-clusters, BBCC-Press identifies very dense, small co-clusters with a clear correlation between the included expression profiles.

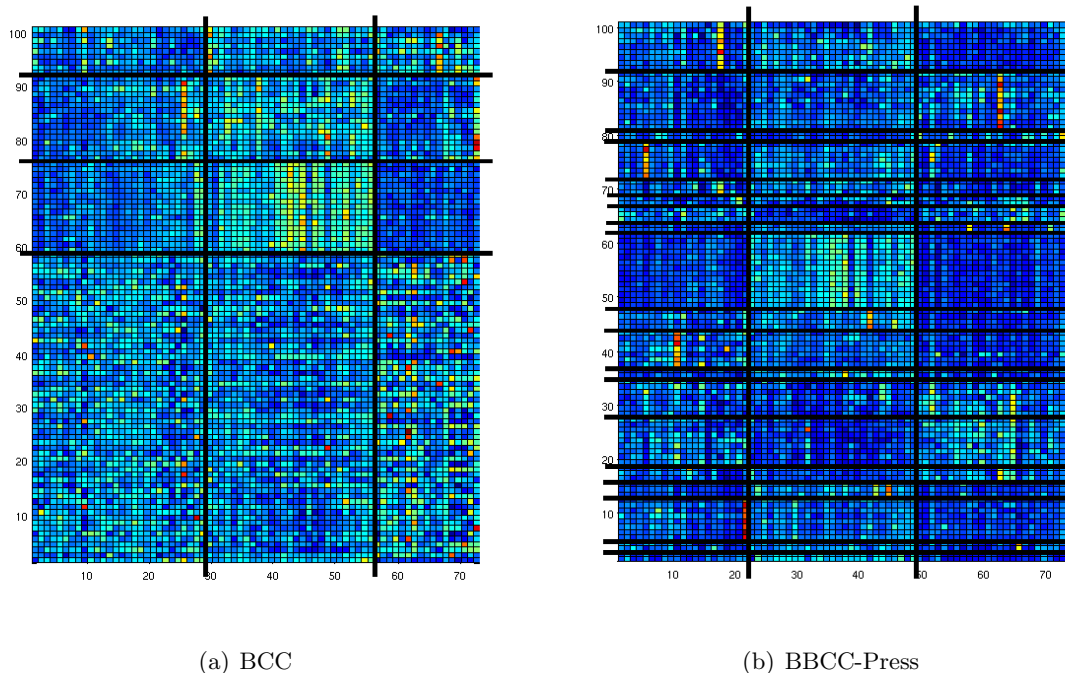


Figure 16: MLL Dataset

#### 10.1.4 P-value Analysis of Co-clusters

Although our main objective in the experiments in Sections 10.1.1 and 10.1.2 was to cluster samples, identifying and investigating clusters of genes is also of interest to biologists. BBCC-

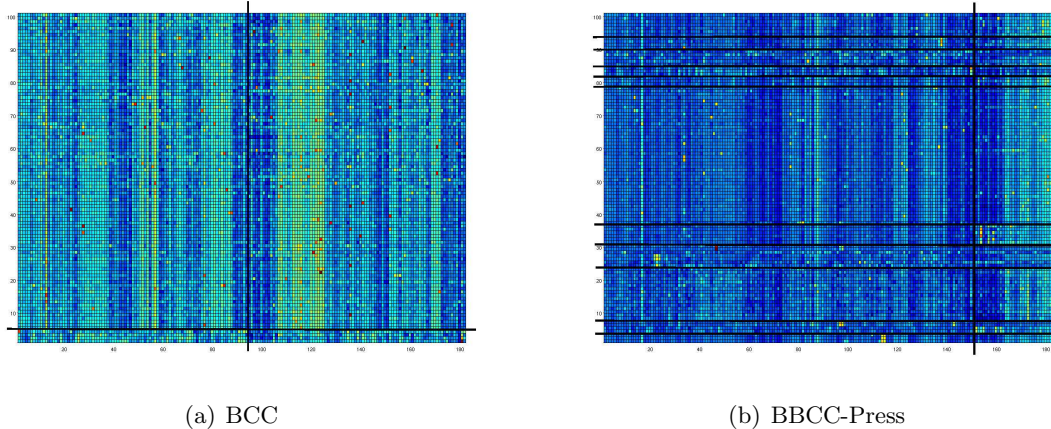


Figure 17: Lung Dataset

Press, by simultaneous pruning and co-clustering of genes and samples identifies gene clusters along with clustering the samples. Here we evaluate the quality of the gene clusters identified by BBCC-Press on the reduced human cancer datasets.

We evaluate the quality of gene clusters by assessing their statistical significance in terms of the p-value. P-value, calculated using the hypergeometric distribution, represents the probability that the intersection of the genes assigned to a cluster with any given functional category occurs by chance. Here we focus on the Gene Ontology (GO) categories within “biological process (BP)”, “cellular component (CC)” and “molecular function (MF)”. Among many publicly available functional profiling tools, we use the DAVID software (<http://david.abcc.ncifcrf.gov>) to compute gene cluster p-values. DAVID adopts the Fisher Exact Test to measure the gene enrichment in annotation terms.

Table 4 illustrates for each gene cluster identified by BBCC-Press on the reduced Leukemia dataset, the GO category with the lowest p-value that the cluster can be mapped to. These results are obtained by clustering 10% of the genes and all the samples of the column standardized dataset. Note that although only 10% of the genes were clustered, the set of background genes for p-value computation includes all the genes in the dataset. One can observe that a very large number of clusters have p-values  $< 0.05$  and hence show statistically significant gene enrichment. BBCC-Press is hence successful at identifying coherent and biologically significant gene clusters along with improving the performance of sample clustering as seen in Sections 10.1.1 and 10.1.2. Due to space limitation, we list only the gene clusters of the Leukemia dataset. The results on the other human cancer datasets were very similar.

## 10.2 Lee Yeast Microarray Dataset

The Lee dataset [LDAM04] consists of the gene expression values of 5612 yeast genes across 591 experiments and can be obtained from the Stanford Microarray Database (<http://genome-www5.stanford.edu/>). Ground truth for evaluating clustering results on this data is based on Gene Ontology (GO) annotations and is available in the form of 121,406 pairwise links between the genes that are known to be functionally related. On this dataset the aim is to find coherent clusters of genes, while pruning away non-discriminative genes and well as experiments. Hence,

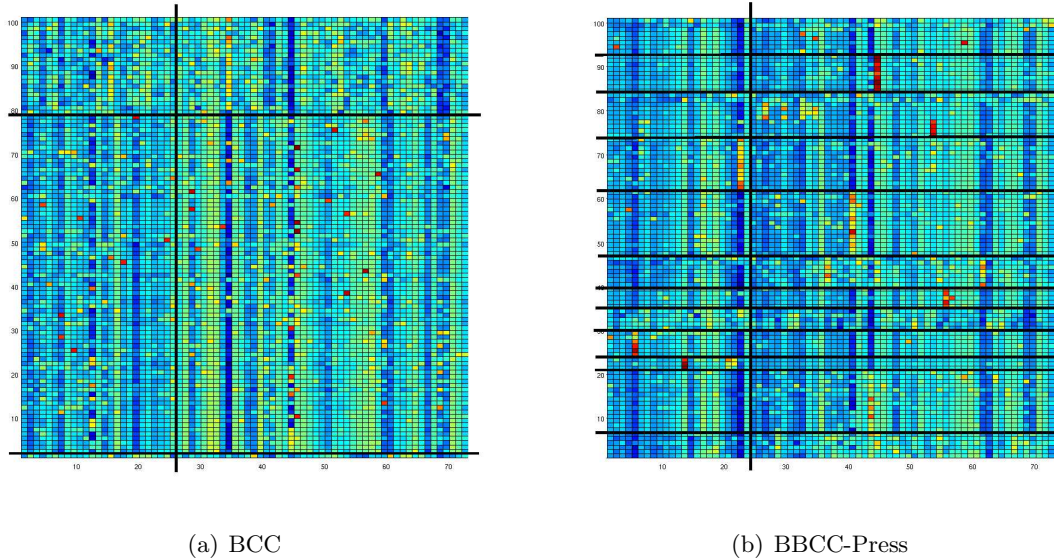


Figure 18: Leukemia Dataset

Cluster	GO Category	cluster size	# genes in category	p-value
1	CC-soluble fraction	42	4	$8.70E - 02$
2	BP-primary metabolism	8	8	$2.20E - 02$
3	CC-extracellular region	15	5	$2.10E - 02$
4	MF-catalytic activity	40	22	$4.60E - 03$
5	BP-regulation of transcription, DNA-dependent	4	3	$7.10E - 02$
6	MF-G-protein coupled receptor activity	43	6	$1.50E - 03$
7	MF-ubiquitin conjugating enzyme activity	15	2	$3.60E - 02$
8	CC-mitochondrion	12	4	$1.30E - 02$
9	MF-structural constituent of ribosome	28	4	$1.80E - 03$
10	BP-reproduction	45	6	$7.30E - 04$
11	BP-neurophysiological process	22	5	$1.30E - 02$
12	BP-cellular lipid metabolism	15	4	$7.80E - 03$
13	BP-DNA repair	5	2	$4.90E - 02$
14	BP-locomotory behavior	12	3	$1.70E - 02$
15	CC-intrinsic to plasma membrane	7	5	$1.60E - 03$
16	CC-endoplasmic reticulum	8	3	$4.40E - 02$
17	CC-intracellular organelle	13	8	$6.30E - 02$

Table 4: Leukemia dataset: p-values of gene clusters identified by BBCC-Press

in this setting BBCC prunes rows as well as columns. The performance of the various clustering algorithms in identifying coherent gene clusters is evaluated using the following measures:

- **Overlap Lift:** Overlap lift measures the likelihood of predicting correct gene links as compared to random chance [GG06].
- **P-value:** The p-value is as defined in Section 10.1.4. The p-values for the yeast gene clusters discovered by the algorithms were obtained using *Funspec* (<http://funspec.med.utoronto.ca/>).

In the Lee dataset, around 15% of the matrix entries are unknown (missing), which makes the clustering problem even more challenging. Figure 19 evaluates the clustering algorithms by comparing the Overlap Lift of the gene clusters at different fractions of the data matrix clustered. The number of genes as well as the experiments clustered are gradually increased from very small values to the entire set and the fraction of the data matrix entries actually assigned to clusters is represented along the X-axis. Since BBC-Press and BC cannot deal with missing data, the missing entries need to be imputed as a preprocessing step. For the results displayed in Figure 19 the missing values are set to zero in the data matrix input to BBC-Press and BC. Note that setting a missing entry to zero is equivalent to assuming that there is no change in the gene expression value. On the other hand, as described in Section 6 the co-clustering algorithms can associate a weight with each matrix entry, which can be set to zero for missing entries. This allows co-clustering based algorithms to ignore missing entries, rather than setting them to a fixed constant value. BCC and BBCC-Press were hence given as input a suitably set weight matrix, indicating the missing entries, for generating the results in Figure 19. BBCC-Press show a dramatic improvement in performance over all the other approaches. This improvement in performance achieved by BBCC-Press is attributed to two factors, co-clustering along with pruning and being able to ignore missing values. In order to evaluate the improvement exclusively due to the bubble co-clustering approach, we also display the performance of BBCC-Press-Zero in Figure 19, where the input to the algorithm is the data matrix with missing entries set to 0. One can observe that the bubble co-clustering alone also brings about significant improvement over BBC-Press and BC.

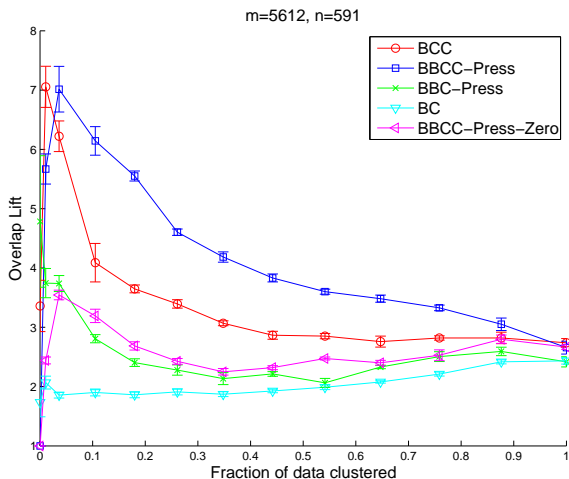


Figure 19: Lee Yeast Dataset - Overlap Lift (II)

Table 5 compares the algorithms with respect to the p-values of the gene clusters obtained with  $k$  set to 20,  $l$  set to 10,  $s_r = 300$  and  $s_c = 400$ . Here in order to level the playing field, all the algorithms are run on the dataset with the missing entries set to 0. The objective is to qualitatively evaluate the improvement in the identified gene clusters brought about by the simultaneous co-clustering and pruning done by BBCC-Press. One can observe that BBCC-Press has the maximum percentage of clusters with low p-values and does very well as compared to BCC and BC.

Algorithm	% of clusters with p-values below		
	$10^{-4}$	$10^{-6}$	$10^{-10}$
BBCC-Press (20 clusters)	65	30	10
BBC-Press (20 clusters)	60	30	10
BCC (13 clusters)	38.5	15.4	7.7
BC (16 clusters)	37.5	18.75	6.25

Table 5: P-value comparison

Many of the gene clusters discovered by BBCC-Press on the Lee data were very pure, comprising of genes belonging to the same biological category. Table 6 displays a few pure, sample clusters and the categories they can be mapped to, with parameter settings  $k = 20$ ,  $l = 10$ ,  $s_r = 300$  and  $s_c = 400$ .

# genes in cluster	# genes in category	category name	p-value
21	14	structural constituent of ribosome [GO:0003735]	$< 10^{-14}$
4	4	cytosolic ribosome (sensu Eukarya) [GO:0005830]	$3.24 \times 10^{-7}$
17	13	nucleobase, nucleoside, nucleotide and nucleic acid metabolism [GO:0006139]	$8.97 \times 10^{-8}$

Table 6: Examples of pure clusters found by BBCC-Press

## 11 Generalized BBCC: Identifying Arbitrarily Positioned Co-Clusters

### 11.1 Motivation

The BBCC algorithm, being an extension of the partitional BCC algorithm suffers from its drawback of being restricted to identifying co-clusters arranged in a rigidly structured grid pattern. However, in real life datasets it is unlikely for co-clusters to be nicely aligned and most often co-clusters are of varying sizes and arbitrarily positioned in the data matrix. Additionally, co-clusters in real data sets e.g. microarray data can be overlapping, which is not taken into consideration by BCC or BBCC. The co-clustering approach proposed by Cheng and Church [CC00] and the plaid model approach [LO02] allows the identification of arbitrarily positioned, overlapping clusters. Both these approaches however identify individual co-clusters sequentially, making them expensive. We now extend the BBCC framework to detect arbitrarily positioned, possibly overlapping co-clusters simultaneously, along with determining the number of co-clusters that exist in the data.

### 11.2 Approach

Generalized BBCC builds up on the BBCC algorithm and inherits its desirable scalability properties. The main idea behind Generalized BBCC is that it uses BBCC-Press to over-partition the data into smaller co-clusters and then hierarchically agglomerates similar, partitioned co-clusters to recover the desired co-clusters. In order to agglomerate co-clusters, we first define a

distance measure between two candidate co-clusters ( $cc1$  and  $cc2$ ) as follows. Let  $cc$  denote the co-cluster formed by the union of the rows and columns in  $cc1$  and  $cc2$ . Compute the co-cluster statistics for  $cc$  based on the chosen scheme to obtain new  $\hat{z}$  values (e.g., in case of scheme 2 compute the mean of all the elements in  $cc$ ). Then compute the average element-wise error  $e$  for  $cc$  as  $e = \frac{1}{N} \sum_{z_{ij} \in cc} d_\phi(z_{ij}, \hat{z}_{ij})$ , where  $N$  is the number of elements in  $cc$ .  $e$  is defined as the distance between  $cc1$  and  $cc2$ . The detailed steps of the algorithm are as follows.

1. Initial co-clustering. Run BBCC-Press with large enough values for the number of row and column clusters ( $k$  and  $l$ ). In particular, if the true number of row and column clusters  $k^{\text{true}}$  and  $l^{\text{true}}$  is known, then set  $k \geq 2 * k^{\text{true}}$  and  $l \geq 2 * l^{\text{true}}$ . Similarly, set the  $s_r$  and  $s_c$  input parameters to sufficiently large values. Since the pruning step (Step 2) takes care of discarding non-coherent co-clusters, setting  $s_r \geq s_r^{\text{true}}$  and  $s_c \geq s_c^{\text{true}}$  is sufficient. Hence, as compared to the BBCC algorithm (Section 6), Generalized BBCC provides more leeway in setting the input parameters. This step identifies  $k \times l$  co-clusters arranged in a grid structure, consisting of  $s_r$  rows and  $s_c$  columns.
2. Pruning co-clusters. The objective of this step is to filter out co-clusters that are not very coherent and have very large error values. A simple and efficient technique to do so is to select the error cut-off value as the point at which the plot of the sorted co-cluster errors shows the steepest rise. Alternatively, a gap statistic based approach can be used to determine the error cut-off. The co-clusters with errors greater than the cut-off are filtered out.
3. Merging similar co-clusters.

This step involves hierarchical, pairwise agglomeration of the co-clusters left at the end of the pruning step (Step 2), to recover the true co-clusters. Each agglomeration identifies the “closest” pair of co-clusters that can be well represented by a single co-cluster model and are thus probably part of the same original co-cluster, and merges them to form a new co-cluster. The rows and columns of the new co-cluster respectively consist of the union of the rows and columns of the two merged co-clusters. Compute all pairwise distances between the current set of co-clusters and choose the pair of co-clusters with a minimum distance for merging. Replace the two merged co-clusters by the new, combined co-cluster. If the total number of co-clusters to be identified is known, stop merging when this number is reached. If not, merge co-clusters until there is a drastic increase in the distance between merged co-clusters. The distance cut-off can be efficiently determined using the same approach used in Step 2 to determine the error cut-off. Note that merging co-clusters by taking a union of the rows and columns allows co-clusters to share rows and columns and hence identifies partially overlapping co-clusters.

A variant of this algorithm can be derived by adopting Ward’s method [War63] to agglomerate co-clusters. According to this approach, in every merge step, the two co-clusters selected to be merged are such that the increase in the co-cluster error after merging them is the least among all candidate pairs of co-clusters. We did not however observe a significant performance difference between the variants of our algorithm using these two different merge criteria.

4. Local Refinement. Use each identified co-cluster in the final set of co-clusters to seed BBCC with  $k = 1$  and  $l = 1$ . This can help each co-cluster to move around locally,

leading to a better solution. This step can be run in parallel for all the co-clusters, to refine all of them simultaneously.

## 12 Generalized BBCC - Experimental Results

We first tested the Generalized BBCC approach in a controlled setting using synthetically generated data (Section 12.1). An advantage of doing this is that synthetic data enables a good evaluation of the proposed approach since true co-cluster labels can be generated for the data matrix entries. For real world microarray datasets, cluster labels are commonly available for gene or experiment clusters, but not for the actual gene-experiment expression values. We also evaluated the ability of Generalized BBCC to find coherent gene clusters on the Lee dataset, details of which are presented in Section 12.2.

### 12.1 Synthetic Data

The synthetic data generation approach follows the steps described in Section 9. The data matrix  $Z$  consists of values randomly selected from a uniform distribution (range 0-10) forming the background, in which blocks of values representing co-clusters are embedded to form a set of co-clusters involving a small fraction of the data. The embedded co-clusters are arbitrarily placed in  $Z$  and are made to overlap in some datasets. The blocks are generated specific to the co-clustering schemes 2 and 6 that we experiment with. The rows and columns of  $Z$  are then randomly permuted. Table 7 describes the synthetic datasets that were used for experimentation.

Data	$m,n$	# co-clusters	sizes of co-clusters	scheme
Dataset 1	500, 500	3	70 by 120, 50 by 50, 160 by 60	2
Dataset 2	500, 200	4	100 by 40, 150 by 40, 70 by 35, 110 by 30	2
Dataset 3	500, 500	3	70 by 120, 50 by 50, 160 by 60	6
Dataset 4	500, 200	4	100 by 40, 150 by 40, 70 by 35, 110 by 30	6

Table 7: Synthetic Datasets

#### 12.1.1 Evaluation metrics

##### 1. Relative non-intersection area (RNIA)

The RNIA metric [PM06] was used to evaluate the ability of the proposed approach to correctly assign cluster labels to the data matrix entries. RNIA compares co-clustering solutions when cluster labels for individual matrix entries are available. Given two co-clustering solutions  $S_1$  and  $S_2$ , the RNIA of the two clusterings is defined as:

$$RNIA(S_1, S_2) = \frac{|U| - |I|}{|U|},$$

where  $|U|$  and  $|I|$  are the number of matrix elements in the union and intersection of the two clusterings respectively. RNIA can be applied to overlapping co-clusters by defining  $|U|$  and  $|I|$  as

$$|U| = \sum_{i,j} \max(n_{ij}^{S_1}, n_{ij}^{S_2}), |I| = \sum_{i,j} \min(n_{ij}^{S_1}, n_{ij}^{S_2}),$$

where  $n_{ij}^S$  is the number of co-clusters in  $S$  that contain the matrix element  $z_{ij}$ .

## 2. Unsupervised cost (UCOST)

This metric was used to evaluate the ability of the co-clustering algorithms as a matrix approximation technique, by computing the quality of the identified co-clusters. The UCOST is defined as the average error between the original matrix entries and their approximation within the corresponding co-clusters, computed across only the entries assigned to co-clusters. Mathematically, the UCOST is computed as follows:

$$UCOST = \frac{\sum_{i=1}^k \sum_{z_{ij} \in C_i} d_\phi(z_{ij} - \hat{z}_{ij})}{\sum_{i=1}^k |C_i|},$$

where  $k$  is the number of identified co-clusters,  $C_i$  is the size of the  $i^{\text{th}}$  co-cluster and  $\hat{z}_{ij}$  is the approximation of  $z_{ij}$  within its corresponding co-cluster. The UCOST depends on the number of co-clusters  $k$  and decreases with a larger value of  $k$  for a given fraction of the data clustered.

### 12.1.2 Results

In this section we display the results obtained by evaluating generalized BBC on the synthetic datasets, based on the visualization of the reconstructed data matrix and comparison of the RNIA and the UCOST measures. For each evaluation method we observed similar trends on all the synthetic datasets and here we illustrate a few of these results.

**Matrix reconstruction.** Figures 20 and 21 display for datasets 2 and 3 the original data matrix and the reconstructed, reordered matrix, where the entries within each identified co-cluster are approximated based on the co-cluster scheme. We can think of the background of the data matrix as being generated from a background model, which is a uniform distribution in this case. The matrix entries not assigned to any cluster by generalized BCC can be used to estimate the parameters of the uniform background distribution. The background of the reconstructed matrix then consists of data points randomly drawn from the background model. The figures show that on both the datasets, the reconstruction is reasonably close to the original dataset. Also, in case of both datasets the algorithm is able to recover the true number of co-clusters.

**RNIA plots.** Figure 22 compares the RNIA for the co-clustering solutions obtained by the BCC, BBCC-Press and Generalized BBCC approaches on each of the synthetic datasets. The X-axis represents different fractions of the data being clustered, as rows and columns are pruned. Note that in case of Generalized BBCC, the actual fraction of the data clustered could be smaller than the corresponding X-axis value, due to the pruning of irrelevant co-clusters. Each plotted value is averaged over 10 randomly initialized runs. On all the datasets the Generalized BBCC approach consistently achieves a significantly lower RNIA than the other approaches.

**UCOST plots.** Figure 23 compares the average approximation error for the co-clusters discovered by the BCC, BBCC-Press and Generalized BBCC approaches on datasets 2 and 3. Since Generalized BBCC discards non coherent co-clusters that have large errors, whereas BCC and BBCC-Press retain all the identified co-clusters, Generalized BBCC has a significantly lower UCOST.

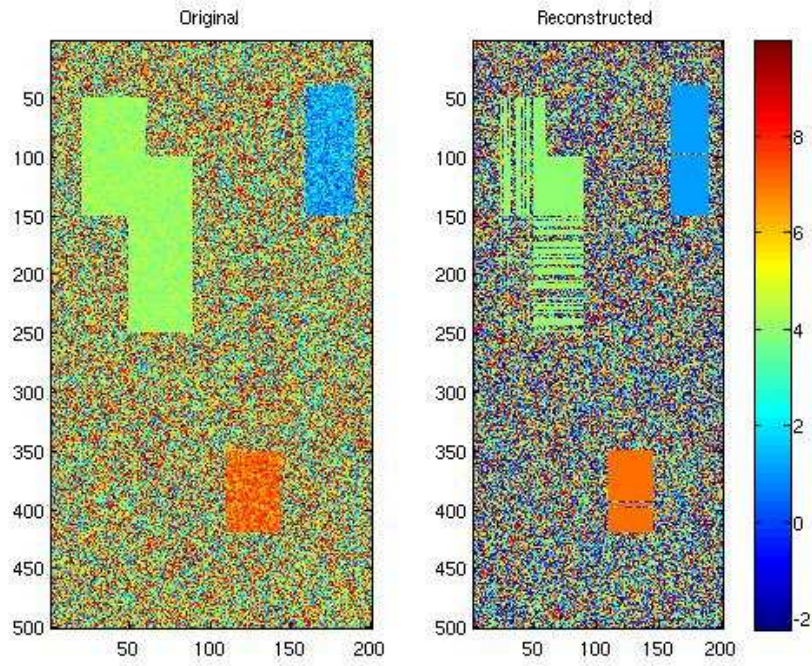


Figure 20: Dataset 2: matrix reconstruction by Generalized BBCC

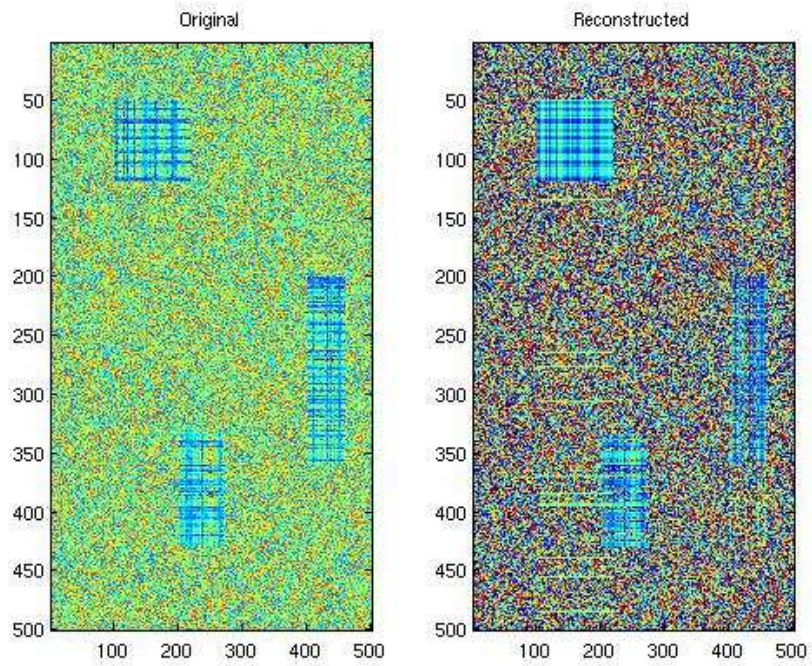
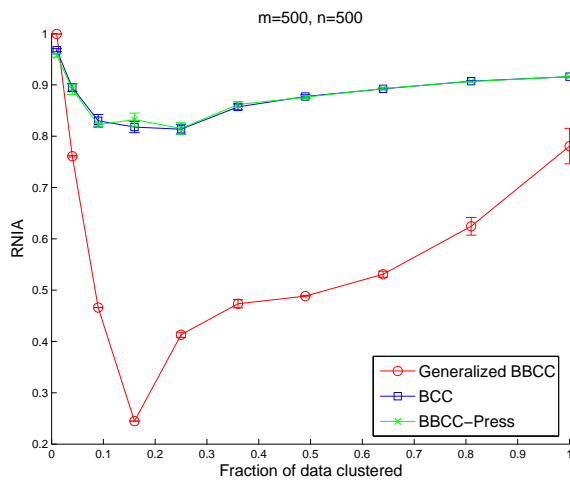
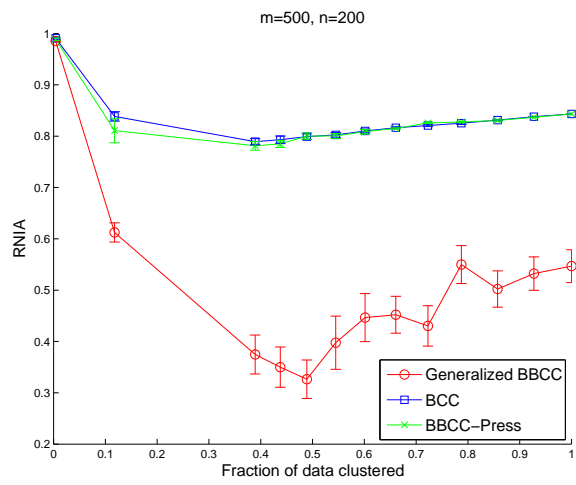


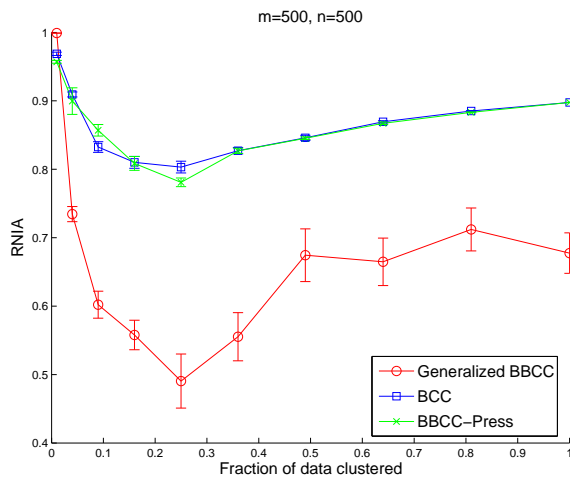
Figure 21: Dataset 3: matrix reconstruction by Generalized BBCC



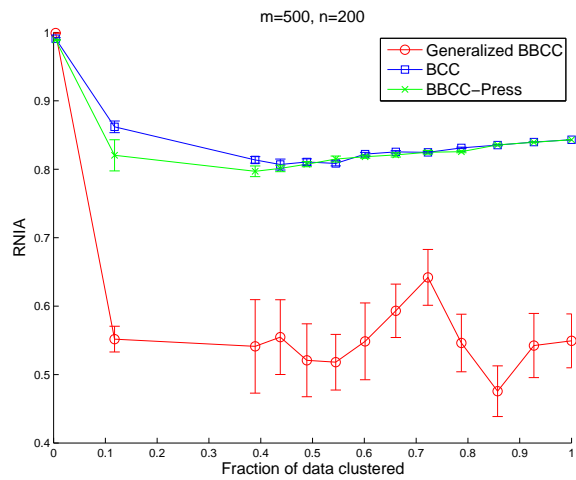
(a) Dataset 1



(b) Dataset 2



(c) Dataset 3



(d) Dataset 4

Figure 22: RNIA Plots comparing co-clustering approaches on synthetic data.

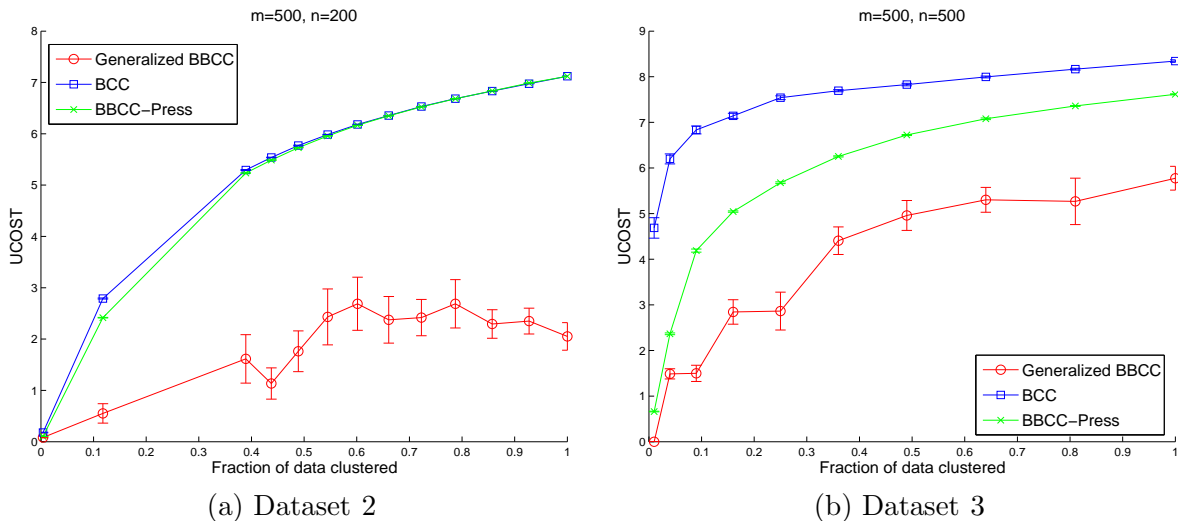


Figure 23: UCOSt Plots comparing co-cluster quality on synthetic data.

## 12.2 Lee Yeast Microarray Dataset

We now evaluate the performance of Generalized BBCC on the Lee dataset described in Section 10.2. Since the ground truth available is only in the form of pairwise gene linkages, we compare the quality of the co-clusters identified by Generalized BBCC, BBCC-Press and BCC by computing the overlap lift for the genes in each co-cluster.

On the Lee dataset, domain knowledge indicates that most of the experiments (columns) are informative, but only a small number of the total set of genes (rows) form coherent clusters. We run the algorithms on the Lee dataset with the input parameters set to  $s_r = 2000$ ,  $s_c = 400$ ,  $k = 50$ ,  $l = 10$ . In our experiments, we set  $s_r$  to a relatively high value to evaluate the ability of Generalized BBCC to discover and discard non-coherent clusters in an unsupervised manner. We can utilize the domain expertise available in this application to guide the model selection performed by Generalized BCC. In the pruning step of Generalized BBCC, we hence prune all but the best 200 co-clusters and then select the best clustering solution after considering upto 50 merge steps. These number are selected given that we want to find the best 150 – 200 co-clusters. We observe that the identified co-clusters are very small as compared to the size of the entire data matrix and also contain a lot of missing values. Hence, running BBCC on the entire dataset with  $k = 1, l = 1$ , seeded with each identified co-cluster is not expected to give very reasonable results, because of which we omit the local refinement step of Generalized BBCC in this case.

Table 8 displays the overlap lift for the different approaches, averaged over 10 trials. The pruning of less coherent co-clusters and merging of similar co-clusters, performed by Generalized BBCC enables it to significantly improve gene cluster quality as compared to BBCC-Press and BCC.

	Generalized BBCC	BBCC-Press	BCC
Overlap Lift	$10.41 \pm 0.71$	$7.36 \pm 0.36$	$5.01 \pm 0.29$

Table 8: Overlap lift results on Lee data

## 13 Conclusions

In summary, we present Bregman bubble co-clustering as a comprehensive framework capable of dealing with several challenges in clustering real life datasets. Our experimental results illustrate that the ability of BBCC to prune away both irrelevant data points and features while simultaneously co-clustering the remaining data enables it to consistently outperform existing approaches including Bregman co-clustering and the one-sided Bregman bubble clustering. Our experiments on the Lee dataset highlight how the ability of BBCC to deal with missing values makes it extremely useful in clustering problems where substantial missing data is an issue. Our analysis of soft BBCC indicates that considering the data as being generated from a mixture of exponential families and a uniform background provides a very natural and intuitive generative model for the data, that can be efficiently estimated by an EM style algorithm. The generalized BCC framework extends BBCC beyond the rigid, grid based co-clustering and enables it to detect arbitrarily positioned, overlapping co-clusters without requiring the number of co-clusters as an input.

The ability of BBCC to operate with different distance measures and detect co-clusters with different structures, along with its scalability and robustness provided by BBCC-Pressurization makes it extremely versatile. While in this paper we focused on clustering microarray data, there are several other domains like market basket analysis and text data clustering where problems due to non-informative data points and features are encountered. It would be worthwhile to investigate the applicability of suitable instances of the BBCC framework to clustering problems in these different domains.

**Acknowledgments:** The research was supported by NSF grants IIS 0325116, IIS 0307792, IIS 0713142, CCF 0431257 and ACI 0093404.

## References

- [ABN<sup>+</sup>99] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. In *Proc. NAS*, volume 96, pages 6745–6750, 1999.
- [AGGR98] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. SIGMOD '98*, pages 94–105, 1998.
- [AM07] D. Agarwal and S. Merugu. Predictive discrete latent factor models for large scale dyadic data. In *Proc. KDD '07*, pages 26–35, 2007.
- [ARD06] M. Al-Razgan and C. Domeniconi. Weighted clustering ensembles. In *SIAM*, pages 258–269, 2006.

- [ASS<sup>+</sup>02] S. A. Armstrong, J. E. Staunton, L. B. Silverman, R. Pieters, M. L. den Boer, M. D. Minden, S. E. Sallan, E. S. Lander, T. R. Golub, and S. J. Korsmeyer. MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia. *Nature Genetics*, 30:41–47, 2002.
- [AWY<sup>+</sup>99] C. Aggarwal, J. Wolf, P. Yu, C. Procopiuc, and J. Park. Fast algorithms for projected clustering. In *Proc. SIGMOD '99*, pages 61–72, 1999.
- [BDCKY03] A. Ben-Dor, B. Chor, R. M. Karp, and Z. Yakhini. Discovering local structure in gene expression data: The order-preserving submatrix problem. *Journal of Computational Biology*, 10(3-4):373–384, 2003.
- [BDG<sup>+</sup>07] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D. Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. *JMLR*, 8:1919–1986, 2007.
- [BFP<sup>+</sup>73] M. Blum, R. Floyd, V. Pratt, R. Rivest, and R. Tarjan. Time bounds for selection. *J. Comput. Syst. Sci.*, 7(4):448–461, 1973.
- [BJ02] T. H. Bø and I. Jonassen. New feature subset selection procedures for classification of expression profiles. *Genome Biology*, 3(4), 2002.
- [BJY04] N. Barakat, J. Jiang, and R. Yu. Bubble agglomeration algorithm for unsupervised classification : a new clustering methodology without a priori information. *Chemometrics and intelligent laboratory systems*, 77:43–49, 2004.
- [CC00] Y. Cheng and G. M. Church. Biclustering of expression data. In *Proc. ICMB '00*, pages 93–103, 2000.
- [CC04] K. Crammer and G. Chechik. A needle in a haystack: Local one-class optimization. In *Proc. ICML '04*, 2004.
- [CD07] H. Cho and I. Dhillon. Effect of data transformation on residue. *UTCS Technical Report TR-07-55*, Downloadable from <ftp://ftp.cs.utexas.edu/pub/techreports/tr07-55.pdf>, 2007.
- [CDGS04] H. Cho, I. S. Dhillon, Y. Guan, and S. Sra. Minimum sum squared residue co-clustering of gene expression data. In *Proc. SDM '04*, 2004.
- [DB02] M. Dettling and P. Bühlmann. Supervised clustering of genes. *Genome Biology*, 3(12), 2002.
- [DF02] S. Dudoit and J. Fridlyand. A prediction-based resampling method for estimating the number of clusters in a dataset. *Genome Biology*, 3(7):research0036.1–0036.21, 2002.
- [DMM03] I. Dhillon, S. Mallela, and D. Modha. Information-theoretic co-clustering. In *Proc. KDD '03*, pages 89–98, 2003.
- [EK SX96] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. KDD '96*, 1996.

- [GG05] G. Gupta and J. Ghosh. Robust one-class clustering using hybrid global and local search. In *Proc. ICML '05*, pages 273–280, 2005.
- [GG06] G. K. Gupta and J. Ghosh. Bregman bubble clustering: A robust, scalable framework for locating multiple, dense regions in data. In *Proc. ICDM'06*, pages 232–243, 2006.
- [GJH<sup>+</sup>02] G. J. Gordon, R. V. Jensen, L. Hsiao, S. R. Gullans, J. E. Blumenstock, S. Ramaswamy, W. G. Richards, D. J. Sugarbaker, and R. Bueno. Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Research*, 62:4963–4967, 2002.
- [GST<sup>+</sup>99] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.
- [Har72] J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129, 1972.
- [HL84] R. A. Harshman and M. E. Lundy. *Research methods for multimode data analysis*, chapter 6. Data preprocessing and the extended PARAFAC model, pages 216–284. New York: Praeger, 1984.
- [KBCG03] Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein. Spectral biclustering of microarray data: Coclustering of genes and conditions. *Genome Research*, 13(4):703–716, 2003.
- [LDAM04] I. Lee, S. Date, A. Adai, and E. Marcotte. A probabilistic functional network of yeast genes. *Science*, 306:1555–1558, 2004.
- [LO02] L. Lazzeroni and A. B. Owen. Plaid models for gene expression data. *Statistica Sinica*, 12(1):61–86, 2002.
- [MO04] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE Trans. on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.
- [NH98] R. M. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. MIT Press, 1998.
- [PHL04] L. Parsons, E. Haque, and H. Liu. Subspace clustering for high dimensional data: a review. *SIGKDD Explor. Newsl.*, 6(1):90–105, 2004.
- [PM06] A. Prikainen and M. Meila. Comparing subspace clusterings. *IEEE Transactions on Knowledge and Data Engineering*, 18(7):902–916, 2006.
- [SLM94] F. C. Sánchez, P. J. Lewi, and D. L. Massart. Effect of different preprocessing methods for principal component analysis applied to the composition of mixtures:

detection of impurities in HPLC-DAD. *Chemometrics and Intelligent Laboratory Systems*, 25(2):157–177, 1994.

[War63] J. Ward. Hierarchical grouping to optimize an objective function. *Journal of American Statistical Association*, 58(301):236 – 244, 1963.