A CLOSE-TO OPTIMUM BI-CLUSTERING ALGORITHM FOR MICROARRAY GENE EXPRESSION DATA

Guojun Li[1*, 3], Qin Ma[1,3], Bingqiang Liu[1,3], Haibao Tang[2], Andrew H. Paterson[2], and Ying Xu[1]

[1]*Computational Systems Biology Laboratory, Department of Biochemistry and Molecular Biology and Institute of Bioinformatics, University of Georgia, Athens, GA 30605. USA*

[2]*Department of Plant Biology, University of Georgia, USA*

[3]*School of Mathematics, Shandong University, China*

[*]*Email: guojun@csbl.bmb.uga.edu*

Motivation: Bi-clustering extends the traditional clustering techniques by attempting to find (all) subgroups of genes with similar expression patterns under to-be-identified subsets of experimental conditions when applied to gene expression data. The bi-clustering strategy has been widely used for analyses of gene expression data and beyond since it was first proposed in 2000 since it provides much increased flexibility and analysis power in identifying co-expressed genes under some but not necessarily all conditions, compared to traditional clustering methods. Still the real power of this clustering strategy is yet to be fully realized due to the lack of effective and efficient algorithms for reliably solving the bi-clustering problem.

Results: We present a novel method to solve a bi-clustering problem, using a (traditional) clustering algorithm combined with a combinatorial technique. The algorithm achieves the same asymptotic computational complexity of the underlying clustering algorithm. While it is a heuristic algorithm in nature, the algorithm achieves close-to optimum classification results across a large collection of benchmark sets.

1. INTRODUCTION

DNA microarrays provide a powerful means for probing the functional states of a cell population by allowing simultaneous observation of mRNA expression patterns of all their genes collected over time and/or under different experimental conditions. By comparing the gene expression patterns collected under different conditions such as cancerous *versus* healthy tissues, one can possibly derive information about genes associated with a particular cellular condition (e.g., cancerous cells at a specific developmental stage). To analyze the complex microarray data, a large number of computational tools have been developed. Among them, clustering of genes based on their similar expression patterns (*co-expressed genes*) using (traditional) clustering strategies[6,18,27] represents one of the most popular techniques for microarray data analyses.

The traditional clustering techniques attempt to, in the context of microarray data analyses, partition a set of genes into "clusters" with similar expression patterns under specified conditions[26] or identify such clusters from an otherwise unstructured microarray dataset[6]. While useful, such clustering algorithms are known to be inadequate for handling the general gene-expression analysis problems, which often need to identify co-expressed genes under some to-be-identified conditions in contrast to finding co-expressed genes under all given conditions. The difficulty in handling the general co-expression identification problem is that for any n given conditions, there are 2^n subsets of conditions to consider, making this general clustering problem much more difficult to solve. The first algorithm for attempting to solve this problem, called a *bi-clustering* problem, was developed by Cheng and Church[5], which attempts to find subsets of conditions under which some (to be identified) subsets of genes have similar expression patterns, using an iterative heuristic strategy. This work inspired further development of more effective bi-clustering algorithms in the past few years, including the work by Kung et al[14]and the work by Li et al. using a Markov Chain Monte Carlo algorithm to search for bi-clusters[15], having led to a number of publicly available

* Corresponding author.

computer servers for bi-clustering analyses of microarray data[22, 24].

A popular way to visualize microarray data for bi-clustering analyses is to represent the dataset as a matrix with rows representing the genes and columns representing the conditions (or the other way around) with each element of the matrix representing the relative mRNA abundance of a gene under a specific condition. Intuitively, a bi-cluster can be identified through row- as well as column-swapping of the original matrix, leading to a sub-matrix in a rearranged matrix consisting of genes with similar expression patterns. When multiple (non-overlapping) bi-clusters are identified through such row and column swappings, the resulting matrix is called to have a "**checker-board structure**" [13, 17].

Formally a *bi-clustering* problem can be defined as to find one or all (possibly overlapping) sub-matrices of a given matrix, each of which shares a pre-defined property over the elements across all its columns (or rows). Each such sub-matrix is called a *bi-cluster*. Examples may include sub-matrices with similar columns (or rows) or sharing more complex column-wise (or row-wise) relationships. One simple example is the one used in[17] in which each column of a to-be-identified sub-matrix should consist of elements of the same numerical value. In this paper, we first present an algorithm for solving a class of bi-clustering problems with the following property: each to-be-identified sub-matrix of a given matrix should have *almost* identical rows, defined in terms of an additive error, and will then explain how to generalize the algorithm to solve more general bi-clustering problems.

Prior to our work, several bi-clustering methods have been developed using combinatorial techniques, such as SAMBA[23]. A central idea of most of these algorithms is to formulate the bi-clustering problem as a maximum balanced bipartite subgraph problem. Though intuitive, this problem formulation is intrinsically computationally intractable, even for a 0-1 matrix, indicating that there are no effective and efficient algorithms for solving the problem. Though our algorithm is a heuristic one, it can generally achieve the optimal bi-clustering results on the test sets of our study, and does so efficiently. The basic idea of the algorithm is to first create a weighted complete graph G with genes represented as vertices and with edges connecting every pair of genes with their weights representing similarities between the corresponding genes' expression patterns. Intuitively, genes in a bi-cluster should induce a heavier subgraph because under the "right" conditions, these genes have the same or highly similar expression patterns. Our goal is to identify the heavy subgraphs in G corresponding to such bi-clusters hidden in the microarray data. The effectiveness of our algorithm lies in our ability to quickly zoom on such conditions that give rise to a bi-cluster in each iteration of the algorithm.

We have assessed the performance of our algorithm on a benchmark set developed by Prelic et al [19], and found that our algorithm performs better than all the popular bi-clustering algorithms, such as ISA[10], BIMAX[19], SAMBA[23] and RMSBE[16]. We have then further applied our algorithm to a number of microarray datasets for cancer type classification, and the results have led to a number of new insights about these cancer microarray data.

2. METHODS

Consider an $n \times m$ matrix M of microarray gene expression data with n genes collected under m conditions, with each gene corresponding to a row and each condition to a column. For the simplicity of discussion, we assume, without loss of generality, that the numerical values of the matrix M are from a finite set (this can be achieved through discretization) Σ with its cardinality $\sigma = |\Sigma|$, and each value is referred to as a *letter*. We assume that the background expression values, i.e. the entries outside the (bi-cluster) sub-matrices of M to be identified, are uniformly and independently distributed over Σ. Define a weighted graph G on M, in which the vertex set V consists of genes and the edge set E consists of all pairs of genes. Each edge, connecting two genes, has a *weight*, defined as the number of common letters in the corresponding positions between the two corresponding rows of M. Throughout the paper, we use "the number of common letters" between two rows (genes) to mean the above. It is known that a special case of this bi-clustering problem, i.e., when M is a binary matrix, is NP-hard through reducing a maximum balanced bipartite subgraph problem to it[23]. Hence our general bi-clustering problem is NP-hard.

Intuitively, a bi-cluster in M corresponds to a "heavier" (connected) subgraph of G compared to an arbitrary subgraph not overlapping such bi-cluster subgraphs, whose total weight is stochastic, and should

in general follow a normal distribution (based on the Central Limit Theorem). Specifically, two genes from the same bi-cluster should have a heavy weight by nature while two arbitrary genes may have a heavy edge only by chance. Our bi-clustering algorithm is built on this observation. It is not difficult to convince ourselves that not all heavy subgraphs represent bi-clusters. The key to effectively solving the bi-clustering problem is to efficiently identify such heavy subgraphs under consistent sets of conditions, i.e., that may correspond to bi-clusters.

Our algorithm iteratively identifies heavy subgraphs as follows. In each iteration, the algorithm starts with the heaviest available edge as the *seed* of a new bi-cluster, labels as the (current) *consensus* the maximal subset of letters common to the two corresponding rows, and then extends it to a bi-cluster with a maximal size by repeatedly adding the next gene whose expression pattern is most consistent with the current consensus and updating the consensus if needed. We do this using every edge as a seed unless the edge has been included in a previously found bi-cluster or deemed to be ineligible as a seed (see the following paragraph); then we pick the one with the largest cardinality as the bi-cluster prediction for the current iteration. The algorithm iterates until no eligible seed is left. Though our algorithm is greedy in nature, it does not in general suffer from the issue of getting stuck in local optima since it considers all possible seeds of a to-be-identified bi-cluster.

The algorithm uses a parameter k as the lower bound on the dimension of the bi-clusters to be identified, possibly provided by the user. In our program, the edges with weights lower than k were first filtered out from the edge set $E(G)$ since they will clearly not be in any to-be-identified bi-clusters. Note that after the filtering step, the graph G may not be complete any more. We assume that the edges of $E(G)$ are given as a sorted list $S=e_1, e_2,\ldots, e_{|E|}$ with $w(e_1) \geqq w(e_2) \geqq \ldots \geqq w(e_{|E|})$. Our algorithm consists of the following three steps:

Step 1 (Seeding): *We maintain a dynamic set S of candidate seeds (edges). Initially S is set to be a sorted list of edges in the decreasing order of weights. In each iteration of the algorithm, we choose the first element of S as the seed, which will be deleted from S after the Expansion step. If the seed is part of a previously found bi-cluster, we will skip it and remove it from S; otherwise it will be used to produce a consensus as follows. Find all the conditions under which the two genes of the seed have identical letters and set this to be the current consensus C. Let V* be the vertex set V after removing the two vertices of the seed. We then find the next gene from V* whose expression pattern is most consistent with C and then update the consensus pattern C taking the new gene into consideration (the updated consensus may change its width). Repeat the above until five genes are included in C, which will be expanded in the Expansion step. We call the corresponding submatrix as the current bi-cluster. The updated consensus size is set to be its current width multiplied by five.*

We use 5 instead of 2 elements as the initial bi-cluster candidate to avoid examining many spurious small "bi-clusters". "5" is determined empirically. In the Expansion step, we use two parameters c and d defined as the column-wise and row-wise conservation levels, respectively. c is defined as the minimum ratio between the number of identical elements in a column and the total number of rows in the current bi-cluster, while the d is defined as the minimum ratio between the number of identical elements between two rows and the length of the current consensus. Let $r=min\{c,d\}$ be the overall conservation level with default value set to be 0.9 (the user can select his/her own value). r is used to deal with the situation of almost identical values in our bi-clustering problem. Throughout our algorithm, we only keep those consensuses whose conservation levels are greater than or equal to r.

Step 2 (Expansion): *We first update the current bi-cluster obtained in the Seeding step according to the parameters c and d. We then expand the current bi-cluster as follows: add a gene from outside of the current bi-cluster whose expression pattern has the highest consistency with the current C, and update the current bi-cluster consistent with the parameters c and d. If C's width is greater than or equal to k, we compare the current bi-cluster to the best one obtained so far using the current seed and then store the one with the larger size. Repeat the procedure until the consensus width is shorter than k or nothing is left from outside of the current bi-cluster. We then retrieve the best bi-cluster for the current seed.*

For each predicted bi-cluster of size $t \times s$, the smaller the size of M, the more significant the bi-cluster is. However, it is difficult to calculate the accurate (statistical) significance of a bi-cluster for the general

case where $t \leqq n$ and $s \leqq m$. Since the significance of a bi-cluster can be easily evaluated when $t=n$, we are able to approximately evaluate the significance of an arbitrary bi-cluster. Let X be a random variable denoting the number of columns with identical letters for a random $t \times m$ matrix defined on Σ. We know that X follows a binomial distribution, i.e., $X \sim B(m, 1/\sigma^{t-1})$. The probability that M has at least s columns with identical letters can be calculated as follows:

$$P_{t,m}(X \geq s) = \sum_{k=s}^{m} C_m^k (\frac{1}{\sigma^{t-1}})^k (1 - \frac{1}{\sigma^{t-1}})^{m-k} \quad (1)$$

In our program, the probability that a submatrix of size at least $t \times s$ occurs in $n \times m$ matrix M was approximated by $(n/t)P_{t,m}(X \geq s)$, i.e.,

$$P_{n,m}(X \geq s, Y \geq t) \approx \frac{n}{t} \sum_{k=s}^{m} C_m^k (\frac{1}{\sigma^{t-1}})^k (1 - \frac{1}{\sigma^{t-1}})^{m-k} \quad (2)$$

where Y is a random variable representing the number of rows of a bi-cluster.

Step 3 (Significance evaluation): *Each identified bi-cluster of size $t \times s$ is output as a candidate when the following*

$$\frac{n}{t} \sum_{k=s}^{m} C_m^k (\frac{1}{\sigma^{t-1}})^k (1 - \frac{1}{\sigma^{t-1}})^{m-k} \quad (3)$$

is lower than a pre-specified threshold (the default is 0.01).

The following gives a pseudo-code of our bi-clustering algorithm that implements the above three steps:

GCLUSTER

Input: data matrix— a discretized microarray data; *k*—a specified lower bound on the width of the to-be-identified bi-clustersr (the default value is 2); *c, d*—conservation rates specified by the user (the default value is 1); *n*—the number of different candidates output by the algorithm; and *P*-value cutoff α.

Output: bi-clusters output in the decreasing order of their significance.

Initialization: Create a weighted graph *G(V, E)* as described above. Sort the edges in *E(G)* as a sequence $S=e_1, e_2, \ldots, e_{|E|}$ such that $w(e_1) \geqq w(e_2) \geqq \ldots \geqq w(e_{|E|})$.

WHILE $i < n$ and $S \neq \phi$

Choose the first element in *S* as a seed.

1) Call the *Seeding* step to calculate the initial consensus pattern with five genes (the current bi-cluster).

2) Call the *Expansion* step to expand the current bi-cluster until the maximum one is reached.

3) Call the *Significance evaluation* step to determine if the maximum bi-cluster output by the Expansion step is statistically significant based on if the value of Eq.(3) is lower than the threshold α.

If (more significant)

output the bi-cluster;

Set $i = i + 1$; and remove the current seed from *S* and continue.

It should be intuitively apparent that the algorithm has the following two distinct features: a) if a significant bi-cluster is being built but not completed in Step 2 due to some reason, leading to a failure of not recognizing the bicluster, this problem could be remedied later with multiple chances by using other edges of the bicluster as seeds. It can find all the statistically significant bi-clusters because any pair of genes in a significant bi-cluster has the opportunity to be a seed and the algorithm always works on the same input data set no mater how many bi-clusters have been output; b) The Expansion step ensures that it always outputs the most significant bi-cluster for each eligible seed, and therefore almost always gets close-to optimum bi-clustering results.

As mentioned in the Introduction section, our algorithm can be extended to solve more general bi-clustering problems, such as finding bi-clusters with rows (or columns) are linearly proportional to each other, i.e., each row can be represented by another row multiplied by some factor. In this case, we only need to redefine the weight of the graph *G* where each edge, connecting two genes, has a *weight*, defined as the maximum number of positions at which the expression patterns of the two genes are linearly proportional.

From the argument of the algorithm, the computational complexity of the algorithm has the same asymptotic complexity to that of its underlying clustering algorithm because each bi-cluster is greedily expanded from a seed just as traditional clustering methods did.

3. RESULTS

We have assessed the performance of our bi-clustering algorithm on several benchmark sets that have been used by previous algorithms, which we now describe. We will then discuss two applications of the algorithm on biological data.

3.1. Tests on synthetic benchmark datasets

To assess the performance of our bi-clustering algorithm GCLUSTER, we first tested it on well-controlled datasets. We applied GCLUSTER to a synthetic benchmark set first used by Prelic et al[19]. Prelic et al. simulated two types of bi-clusters – 'constant' bi-clusters and 'coherent' bi-clusters[17], where 'constant' bi-clusters refer to sub-matrices containing identical values for all entries, while the more general model – 'coherent' bi-clusters are sub-matrices with values identical in each column but varying across the columns[19]. Both problems are solvable by GCLUSTER since our algorithm is based on the more general 'coherent' model.

The benchmark set was generated by *implanting* bi-cluster matrices into a larger background matrix. When implanting a bi-cluster matrix, the values of the bi-cluster matrix were used to replace the value in the implanted location in the background matrix, maintaining the property that elements from the same row (and column) of the bi-cluster matrix are on the same row (and column) in the implanted matrix. Under 'constant' and 'coherent' models, Prelic's benchmark can be used to compare the performance of bi-clustering algorithms considering the two scenarios: 1) matrices with varying levels of noise and 2) matrices with varying degrees of overlap among the bi-clusters in the same background matrix. The whole benchmark set comprises four sets of data.

The sub-matrices were implanted into the background matrices whose values follow normal distributions with varying standard deviations σ – used to model the different level of 'noise' (scenario 1) and the level of overlaps among the bi-clusters (scenario 2). In scenario 1, ten non-overlapping bi-clusters of size 10(genes)×5(conditions) were implanted into background matrices of size 100×50 while the level of background noise (controlled by σ) range from 0 to 0.25 for the 'constant' model (Figure 1A) and 0 to 0.10 for the 'coherent' model (Figure 1B). In scenario 2, the background noise parameter σ was 0, and the bi-clusters with size $(10+d)\times(10+d)$ were implanted into a $(100+d)\times(100+d)$ matrix at the interval of 10 genes and 10 conditions, thereby forcing the bi-clusters to be overlapping with each other at different levels (controlled by d) for both the 'constant' (Figure 1C) and 'coherent' (Figure 1D) models. Further details about construction of the benchmark sets can be found in reference[19].

For comparative studies between our algorithm and the previous ones, we did not include three earlier bi-clustering algorithms, Cheng-Church method (CC)[5], xMotif and OPSM in our study, because they were shown to have fairly low performance accuracy (below 50%) in recovering implanted bi-clusters by previous studies[16, 19]. Three algorithms, BIMAX[19], Iterative Signature Algorithm (ISA)[9], and SAMBA[23], achieved relatively good performance therefore we compared the performance of GCLUSTER with these methods. We used the BIMAX and ISA algorithms implemented in BICAT[3] and the SAMBA algorithm implemented in EXPANDER[21]; both software packages are publicly available. In addition, we also included a recently published bi-clustering algorithm RMSBE based on mining maximum-similarity bi-clusters[16]. The parameters for running these bi-clustering algorithms were taken either from their default settings or following the parameters suggested by the original authors (see supplementary information on our website). Pre-processing and post-processing were performed in a consistent manner with the previous benchmark study[19].

We first compared the bi-clustering results for scenario 1. Surprisingly, we found that the most recent method, RMSBE shows the poorest performance (at an accuracy level below 80%) among the five tested algorithms. This was also noted by Wu, et al.[25] possibly because RMSBE is not appropriate for the situations where the noise levels within bi-clusters and the background are very similar. The other four algorithms except for BIMAX identified all the implanted bi-clusters for the 'constant' model, as shown in Figure 1A. From Figure 1B, for the 'coherent' case, we can see that ISA has the best performance among the five programs while GCLUSTER consistently ranks the second after ISA. The performances of all these algorithms are reduced proportionally to the level of noise in the background matrix. In the test case with the most 'noise' (σ=0.10, 'coherent' model), the 90% accuracy by GCLUSTER is lower than 98% of ISA, but is better than both BIMAX and SAMBA with 84% and 80% accuracy, respectively. Then we considered situations where there are overlapping clusters – scenario 2 (Figure 1C, 1D). RMSBE continues to show relatively low accuracy in recovering the implanted bi-clusters compared to the other four programs. The performances of SAMBA and ISA are affected by the presence of overlapping bi-clusters. Specifically, as the overlap between bi-clusters increases, the performances of both

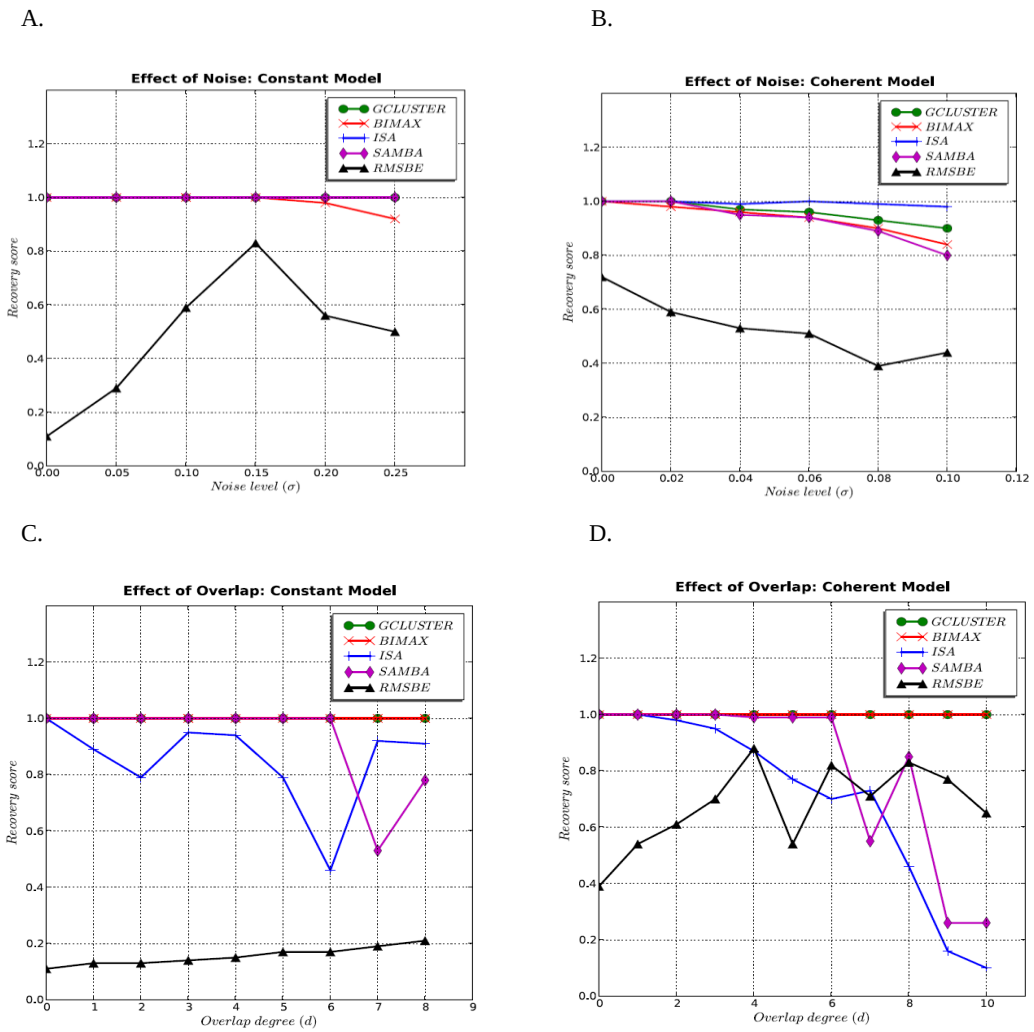

Fig. 1. Comparison of recovery accuracy of GCLUSTER with four other bi-clustering algorithms on the Prelic benchmark [9]. The analysis reveals both the effects of increasing noise levels (scenario 1) for 'constant' (A) and 'coherent' (B) models and varying degrees of overlapping (scenario 2) for 'constant' (C) and 'coherent' (D) models. Note that the recovery score is calculated similarly to (Prelic, et al., 2006) using

$$S_G^*(M_{opt}, M) = \frac{1}{|M_{opt}|} \sum_{G_{opt} \in M_{opt}} \max_{G \in M} \frac{|G_{opt} \cap G|}{|G_{opt} \cup G|},$$

where M_{opt} is the set of implanted bi-clusters; M is the set of recovered bi-clusters; G stands for genes sets within the bi-cluster.

programs drop substantially and the extent of the performance drop is correlated with the degree of overlap *d*. On the same datasets, neither GCLUSTER nor BIMAX is affected by the increasing degree of overlap as we see from the same figure that both methods have identified all the overlapping bi-clusters. Indeed, the ISA method, while performing very well in scenario 1, has the worst performance when bi-clusters overlap, in some cases (*d*=10, 'coherent' model) suffering a 90% performance reduction compared to GCLUSTER/BIMAX.

Overall on the Prelic datasets, we found that GCLUSTER has consistently performed in the best in the most general case. It appears that as though ISA has the marginal advantage (up to 8%) over GCLUSTER on the 'noisy' case, its performance drops up to 90% when the bi-clusters overlap.

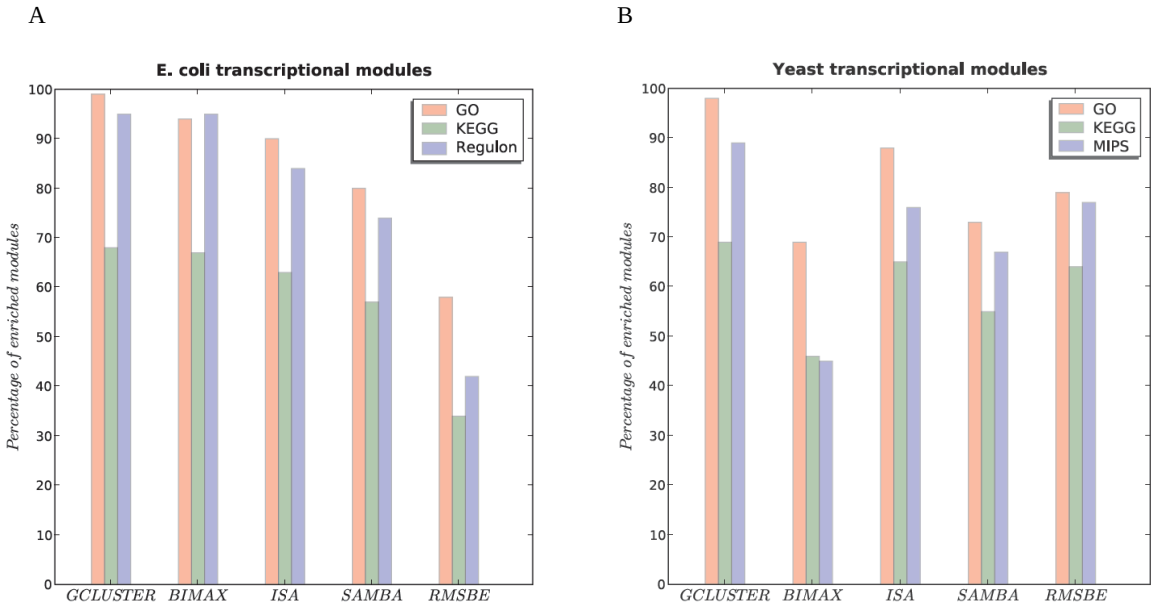

Fig. 2. Evaluation and comparison of different bi-clustering algorithms on *E. coli* and yeast microarray data. (A) Proportions of *E. coli* bi-clusters that have significant overlap (p<0.01) with GO biological processes, KEGG pathways, and experimentally verified regulons. (B) Proportions of yeast bi-clusters that are statistically enriched (p<0.01) in GO biological processes, KEGG pathway and MIPS functional catalog.

3.2. Tests on global transcriptional datasets

We now compare and evaluate the aforementioned algorithms on global microarray data collected on two different organisms (*E. coli* and yeast). When analyzing the whole transcriptome microarray data, one challenging problem is to find the "transcriptional modules", which represent modular components in the (global) gene regulatory network, defined as a set of tightly co-regulated genes along with a set of associated conditions that trigger the co-regulation[10], making it a natural application problem for the bi-clustering methods. It is known that some transcriptional modules show co-regulations only under a narrow range of conditions and have weak global correlations among their gene expression patterns, therefore not easily detectable by the traditional clustering methods. In addition, some transcriptional modules may overlap due to the combinatorial regulation by multiple transcriptional factor[10], which would also complicate the use of the traditional clustering techniques. The goal of this exercise is to test the effectiveness of the bi-clustering algorithms in identifying such transcriptional modules.

Our first test case includes the microarray gene expression data for 4217 *E. coli* genes collected under 264 conditions from the M3D database (*E. coli* array version 4 build 3)[7]. The values in the original microarray dataset are log-2 values of the fluorescence intensities. As a pre-processing step, we centered these values by subtracting the median for each gene so that each entry was transformed to log-2 ratio with respect to the median intensity. We then transformed this 4217 × 264 matrix to a simplified matrix with three distinct values, -1, 0, 1 as follows. For each column in the matrix, the values of the top 5 percentile of the most up-regulated genes were converted to 1 (up-regulated) and the values of the 5 percentile of the most down-regulated genes were converted to -1 (down-regulated) and the rest were assigned 0 (unchanged). The goals of our analysis is to identify bi-clusters hidden in the microarray data, and study their relationships to biological pathways, as defined in terms of biological processes by the GO functional classification scheme[2]. In addition, we have also considered two other functional classification schemes, namely KEGG

pathways[11] and experimentally validated regulons from the *EcoCyc* database [12].

For each identified bi-cluster, we calculate the *p*-value using Fisher's exact test as defined in GeneMerge program[4] as follows: suppose we have a functional classification that partitions the total N genes into k classes $C_1, \ldots, C_k$. Let B be a bi-cluster of n genes, with n_j genes belonging to class C_j. The *p*-value of B can be calculated as

$$p(B) = \min_{j \in 1..k} \left(\frac{\binom{|C_j|}{n_j}\binom{N-|C_j|}{n-n_j}}{\binom{N}{n}} \right) \qquad (4)$$

which essentially measures the statistical significance of the functional enrichment by B's most dominating functional class of genes, i.e., genes in the same biological process. Clearly the smaller the *p*-value of a B is, the more likely that B's genes are from the same biological process.

We have run the five bi-clustering algorithms like before on this dataset. For each algorithm, we calculate the proportions of bi-clusters that have significant *p*-values (below a pre-selected *p*-value cutoff) and compare these sub-matrices as a way to compare their performance. To facilitate better comparisons among the bi-clustering results from different algorithms, we applied a procedure following Prelic et al.[19] to remove the substantially overlapped bi-clusters so that no two bi-clusters overlap more than 25% of their sizes. In addition, we restrain our comparisons to the 100 best scoring bi-clusters for each algorithm.

Among the five tested algorithms, GCLUSTER consistently show the highest enrichment based on the three functional classifications, with BIMAX ranking the second after GCLUSTER on this dataset. Specifically, 99% of the GCLUSTER bi-clusters show substantial enrichment with GO biological processes, 95% of the GCLUSTER results show significant overlap (p<0.01) with known regulons, and 68% enriched in KEGG pathways[11], while the detailed comparisons with other programs are given in Figure 2A.

As our second test, we used yeast (*S. cerevisiae*) microarray data. The test data is derived from a study of transcriptional responses of 2993 genes under 173 different stress conditions[8]. This dataset has been used to validate bi-clustering algorithms in several previous studies[16, 19]. The data entries represent log-2 test-to-reference ratios in the dual-channel chips, and are already normalized so there is no need for further pre-processing. Similar to the *E. coli* data analysis, we evaluated each bi-cluster generated by different bi-clustering algorithms in terms of functional enrichments based on GO biological processes, MIPS yeast functional catalog[20] and KEGG pathways (Figure 2B). From Figure 2B, we can see that GCLUSTER has the highest functional enrichment among the five tested algorithms.

Through the above comparative analyses on the performance of five bi-clustering algorithms on the two sets of microarray data, we have shown that GCLUSTER is capable of revealing high quality bi-clusters in both prokaryotic and eukaryotic expression profiles, and the genes within the bi-clusters show good correlations with known functions and pathways. This study thus suggests the potential of extracting and applying the sub-structures in the global expression data when annotating metabolic pathways and regulatory networks. The combination of microarray-based bi-clusters and empirical knowledge of shared functional groups or regulatory elements would allow for more accurate detection of transcriptional modules.

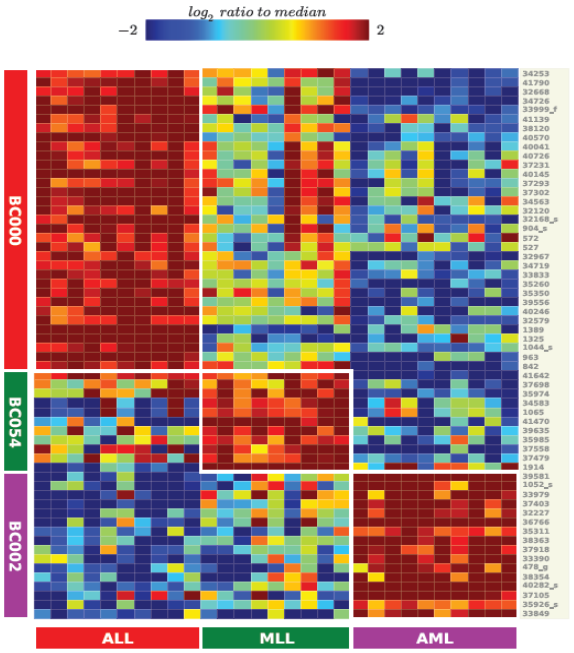

Fig. 3. Visualization of three bi-clusters (BC000, BC002, BC054), which were selected based on the specificity to certain subtype of leukemia (ALL/MLL/AML). The IDs shown to the right of the heat-map are Affymetrix probe IDs.

3.3. Identifying signatures for cancer subtyping

We now extend the application of our bi-clustering algorithm to the problem of cancer subtype classification. The basis of this analysis is that we expect that some pathways unique to each cancer subtype may get activated across the majority of the patients of this cancer subtype, and hence the activation of the genes in these pathways can be possibly used as a signature for cancer subtyping. By finding such activated gene groups for each cancer type, we can possibly do cancer classification based on their molecular signatures. Apparently this problem could be formulated as a bi-clustering problem on microarray gene expression data. Actually, there have been several studies that used bi-clustering as part of a larger analysis pipeline to do cancer subtyping[13].

We have used the leukemia data collected by Armstrong et al[1] and searched for bi-clusters that might be characteristic of different leukemia subtypes (ALL, MLL and AML). This dataset consists of 12,533 probes from 72 patients of different subtypes of leukemia (44 ALL, 20 MLL and 28 AML patients, respectively), which were produced on Affymetrix U95A oligo-nucleotide arrays. We did the same pre-processing on the array data as we have done on the previous test case, and then carried out bi-clustering analyses on the transformed matrix consisting of three different values, -1, 0, and 1 like before.

Using GCLUSTER, we have identified a total of 463 bi-clusters in the dataset (outputs available on our website). We made the following observations about the predicted bi-clusters: 5 bi-clusters contain samples from only one cancer subtype, 121 bi-clusters have samples from two subtypes and 338 bi-clusters from all three subtypes. Although only 5 bi-clusters were found to have specificity for a particular sub-type, these bi-clusters are highly significant and distinct. Figure 3 gives an example of three selected bi-clusters that each shows subtype-specificity (BC000, BC002, BC054; with *p*-values 6.3e-154, 7.2e-84, 5.5e-38 respectively). In this example, GCLUSTER identifies the classical 'checker-board' sub-structures inside the original microarray data, where the three selected bi-clusters each corresponds to a particular leukemia sub-type, with BC000 specific to ALL, BC054 specific to MLL and BC002 specific to AML. It remains interesting for future testing how the genes within these bi-clusters are related and how they establish the cancer sub-type as a unique entity.

The bi-clusters that contain samples between two or more sub-types are probably clinically as informative as the subtype-specific bi-clusters. For example, we have found that among the resulting bi-clusters, a few bi-clusters (e.g. BC005, BC006 etc.) show opposite trend for different ALL and AML. In particular within bi-cluster BC005, samples from ALL patients are all up-regulated while samples from AML patients are all down-regulated; BC006 show exactly the opposite pattern where ALL samples are down-regulated and AML samples are up-regulated. These bi-clusters would contain candidates of selectively expressed genes for needed molecular targets. Note that this was not possible using some other bi-clustering algorithms such as BIMAX, since BIMAX only deals with binary discretizations (change vs. no- change)[19] as opposed to multi-class discretizations (up-regulated, no-change and down-regulated in our analysis).

As result of bi-clustering on the cancer data, we have shown that GCLUSTER is capable of uncovering genes that uniquely characterize or differentiate specific clinical groups. Future work could be focused on refining the subtype-specific bi-clusters, based on which we can further integrate into more accurate supervised classification pipeline for cancer diagnostics and classification problems.

Acknowledgments

This research was supported in part by National Science Foundation (#NSF/DBI-0354771, #NSF/ITR-IIS-0407204, #NSF/DBI-0542119, and #NSF/CCF-0621700), by a "distinguished scholar" grant from Georgia Cancer Coalition and by the U.S. Department of Energy's BioEnergy Science Center (BESC) grant through the Office of Biological and Environmental Research. GJ Li's work was supported in part by grants (60673059, 10631070 and 60373025) from NSFC and the Taishan Scholar Fund from Shandong Province, China. We also thank Dongsheng Che and Kun Xu for their help and insightful discussions on the work.

References

1. Armstrong, S.A., Staunton, J.E., Silverman, L.B., Pieters, R., den Boer, M.L., Minden, M.D., Sallan, S.E., Lander, E.S., Golub, T.R. and Korsmeyer, S.J.

(2002) MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia, *Nature genetics*, 30, 41-47.

2. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M. and Sherlock, G. (2000) Gene ontology: tool for the unification of biology. *The Gene Ontology Consortium*, *Nature genetics*, 25, 25-29.
3. Barkow, S., Bleuler, S., Prelic, A., Zimmermann, P. and Zitzler, E. (2006) BicAT: a biclustering analysis toolbox, *Bioinformatics* (Oxford, England), 22, 1282-1283.
4. Castillo-Davis, C.I. and Hartl, D.L. (2003) GeneMerge--post-genomic analysis, data mining, and hypothesis testing, *Bioinformatics* (Oxford, England), 19, 891-892.
5. Cheng, Y. and Church, G.M. (2000) Biclustering of expression data, Proceedings / ... *International Conference on Intelligent Systems for Molecular Biology ; ISMB*, 8, 93-103.
6. Eisen, M.B., Spellman, P.T., Brown, P.O. and Botstein, D. (1998) Cluster analysis and display of genome-wide expression patterns, *Proc Natl Acad Sci* U S A, 95, 14863-14868.
7. Faith, J.J., Driscoll, M.E., Fusaro, V.A., Cosgrove, E.J., Hayete, B., Juhn, F.S., Schneider, S.J. and Gardner, T.S. (2007) Many Microbe Microarrays Database: uniformly normalized Affymetrix compendia with structured experimental metadata, Nucleic acids research.
8. Gasch, A.P., Spellman, P.T., Kao, C.M., Carmel-Harel, O., Eisen, M.B., Storz, G., Botstein, D. and Brown, P.O. (2000) Genomic expression programs in the response of yeast cells to environmental changes, *Molecular biology of the cell*, 11, 4241-4257.
9. Ihmels, J., Bergmann, S. and Barkai, N. (2004) Defining transcription modules using large-scale gene expression data, *Bioinformatics*, 20, 1993-2003.
10. Ihmels, J., Friedlander, G., Bergmann, S., Sarig, O., Ziv, Y. and Barkai, N. (2002) Revealing modular organization in the yeast transcriptional network, *Nat Genet*, 31, 370-377.
11. Kanehisa, M. (2002) The KEGG database, Novartis Foundation symposium, 247, 91-101; discussion 101-103, 119-128, 244-152.
12. Keseler, I.M., Collado-Vides, J., Gama-Castro, S., Ingraham, J., Paley, S., Paulsen, I.T., Peralta-Gil, M. and Karp, P.D. (2005) EcoCyc: a comprehensive database resource for Escherichia coli, *Nucleic acids research*, 33, D334-337.
13. Kluger, Y., Basri, R., Chang, J.T. and Gerstein, M. (2003) Spectral biclustering of microarray data: coclustering genes and conditions, *Genome Res*, 13, 703-716.
14. Kung, S.Y., Mak, M.W. and Tagkopoulos, I. (2006) Symmetric and asymmetric multi-modality biclustering analysis for microarray data matrix, *Journal of bioinformatics and computational biology*, 4, 275-298.
15. Li, H., Chen, X., Zhang, K. and Jiang, T. (2006) A general framework for biclustering gene expression data, *J Bioinform Comput Biol*, 4, 911-933.
16. Liu, X. and Wang, L. (2007) Computing the maximum similarity bi-clusters of gene expression data, *Bioinformatics* (Oxford, England), 23, 50-56.
17. Madeira, S.C. and Oliveira, A.L. (2004) Biclustering algorithms for biological data analysis: a survey, *IEEE/ACM transactions on computational biology and bioinformatics / IEEE, ACM*, 1, 24-45.
18. McLachlan, G.J., Bean, R.W. and Peel, D. (2002) A mixture model-based approach to the clustering of microarray expression data, *Bioinformatics*, 18, 413-422.
19. Prelic, A., Bleuler, S., Zimmermann, P., Wille, A., Buhlmann, P., Gruissem, W., Hennig, L., Thiele, L. and Zitzler, E. (2006) A systematic comparison and evaluation of biclustering methods for gene expression data, *Bioinformatics*, 22, 1122-1129.
20. Ruepp, A., Zollner, A., Maier, D., Albermann, K., Hani, J., Mokrejs, M., Tetko, I., Guldener, U., Mannhaupt, G., Munsterkotter, M. and Mewes, H.W. (2004) The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes, *Nucleic acids research*, 32, 5539-5545.
21. Shamir, R., Maron-Katz, A., Tanay, A., Linhart, C., Steinfeld, I., Sharan, R., Shiloh, Y. and Elkon, R. (2005) EXPANDER--an integrative program suite for microarray data analysis, *BMC bioinformatics*, 6, 232.
22. Sheng, Q., Moreau, Y. and De Moor, B. (2003) Biclustering microarray data by Gibbs sampling, *Bioinformatics*, 19 Suppl 2, ii196-205.
23. Tanay, A., Sharan, R. and Shamir, R. (2002) Discovering statistically significant biclusters in gene expression data, *Bioinformatics*, 18 Suppl 1, S136-144.

24. Wu, C.J. and Kasif, S. (2005) GEMS: a web server for biclustering analysis of expression data, *Nucleic Acids Res*, 33, W596-599.
25. Wu, Z., Ao, J. and Zhang, X. (2007) Finding distinct biclusters from background in gene expression matrices, *Bioinformation*, 2, 207-215.
26. Xu, Y., Olman, V. and Xu, D. (2002) Clustering gene expression data using a graph-theoretic approach: an application of minimum spanning trees, *Bioinformatics*, 18, 536-545.
27. Yeung, K.Y., Medvedovic, M. and Bumgarner, R.E. (2003) Clustering gene-expression data with repeated measurements, *Genome Biol*, 4, R34.