Gene Expression Module Discovery Using Gibbs Sampling

Chang-Jiun Wu[1*] **Yutao Fu**[1*]
terrence@bu.edu bibin@bu.edu
T. M. Murali[2] **Simon Kasif**[1,3]
murali@cs.vt.edu kasif@bu.edu

[1] Boston University Bioinformatics Program, Boston, MA 02215, USA
[2] Department of Computer Science, Virginia Polytechnic Institute andState University, Blacksburg, VA24061, USA
[3] Department of Bioengineering, Boston, MA 02215, USA

Abstract

Recent advances in high throughput profiling of gene expression have catalyzed an explosive growth in functional genomics aimed at the elucidation of genes that are differentially expressed in various tissue or cell types across a range of experimental conditions. These studies can lead to the identification of diagnostic genes, classification of genes into functional categories, association of genes with regulatory pathways, and clustering of genes into modules that are potentially co-regulated by a group of transcription factors. Traditional clustering methods such as hierarchical clustering or principal component analysis are difficult to deploy effectively for several of these tasks since genes rarely exhibit similar expression pattern across a wide range of conditions. Bi-clustering of gene expression data is a promising methodology for identification of gene groups that show a coherent expression profile across a subset of conditions. This methodology can be a first step towards the discovery of co-regulated and co-expressed genes or modules. Although bi-clustering (also called block clustering) was introduced in statistics in 1974 few robust and efficient solutions exist for extracting gene expression modules in microarray data. In this paper, we propose a simple but promising new approach for bi-clustering based on a Gibbs sampling paradigm. Our algorithm is implemented in the program GEMS (Gene Expression Module Sampler). GEMS has been tested on synthetic data generated to evaluate the effect of noise on the performance of the algorithm as well as on published leukemia datasets. In our preliminary studies comparing GEMS with other bi-clustering software we show that GEMS is a reliable, flexible and computationally efficient approach for bi-clustering gene expression data.

Keywords: bi-clustering, gene expression, Gibbs sampling, microarray, module.

1 Introduction

Recent advances in high throughput profiling of gene expression allow simultaneous measurements of the abundance of mRNA molecules on a genome-wide scale. As an increasing number of large-scale microarray experiments are carried out, analysis of the expression data produced by these experiments remains a major challenge. A key step in the analysis of microarray data is the identification of groups of genes that exhibit similar expression patterns. Canonical clustering methods, such as hierarchical clustering [3], K-means [7], or SOM (self-organizing maps) [16] assume that genes in a cluster behave similarly over all the conditions. These clustering methods produce reliable results for microarray experiments performed on homogeneous conditions. However, this assumption is no longer appropriate when the conditions of an experiment vary greatly. In contrast with classical clustering that reveals

*These two authors contributed equally to this paper. Correspondence should be sent to terrence@bu.edu genes that behave similarly over all the conditions, bi-clustering of expression data seeks a subset of genes that are expressed homogeneously across a subset of conditions. The conserved gene subsets are often regulated by similar mechanisms or by certain sets of transcriptional factors. We refer to groups of coexpressed genes as *gene expression modules.* Our *gene expression modules* are candidates for further screening required for regulatory module identification, e.g., by ChIP technologies or promoter identification.

Exhaustively searching for gene expression modules is computationally difficult because of the large number of possible combinations of genes and samples to examine. Hartigan's block clustering algorithm attempts to repeatedly rearrange the rows and columns of the matrix of gene expression values so that the rearranged matrix contains several disjoint blocks (contiguous sub-matrices) of highly correlated values [6, 18]. Cheng and Church used a greedy bi-clustering approach and were the first to apply it to gene expression data [2]. Getz *et al.* devised a coupled two-way iterative clustering algorithm to analyze gene microarray data [4]. Lazzeroni and Owen introduced a plaid model using a form of overlapping two-sided clustering with an embedded ANOVA in each layer [10]. Califano *et al.* applied a pattern-discovery algorithm called SPLASH initially developed for finding patterns in strings to gene expression data [1]. Tanay *et al.* combined bipartite graph theoretic and statistical considerations and developed the SAMBA algorithm to reduce the bi-clustering problem to that of finding heaviest subgraphs in a bipartite graph [17]. Kluger *et al.* use a spectral bi-clustering method to find a checkerboard structures in expression matrices. The eigenvectors corresponding to characteristic expression patterns across genes or conditions can be found by linear algebra techniques [8]. Segal *et al.* construct "regulatory programs" from DNA microarray data to generate hypotheses of the form "regulator X regulates gene Y under condition Z". A regulatory program specifies a gene module that includes the regulatory genes that control the module, and the mRNA expression profiles of the module's regulates as a boolean function of the expression of the module's regulators [13].

In this paper we develop a new approach to bi-clustering based on Gibbs sampling. Gibbs sampling is a special Markov Chain Monte Carlo scheme and has been deployed successfully in the discovery of motifs in DNA and protein sequences [9]. Compared to local search techniques or approaches based on expectation maximization, Gibbs sampling uses a probabilistic sampling scheme to escape from local optima. Gibbs sampling has been also implemented by Sheng *et al.* and applied to bi-clustering microarray data [14]. They first discretized the expression values of each gene in the microarray data into fixed number of bins. They observed that finding a conserved module in this discretized dataset resembles the problem of finding motif subsequences in sequence data. This analogy was also previously observed by the authors of SPLASH [1].

In this paper, we propose a simpler Gibbs sampling scheme and expand its application to bi-clustering continuous gene expression data. Before attempting to describe the algorithm, it is useful to consider first the desired properties of a gene expression module. Genes in a module should be expressed homogeneously across the subset of samples, that is, the expression values should be limited to a small range. Computing one module for a small number of samples makes the representation over-specific. Therefore, we desire that each module should be matched by a large fraction of the samples in the dataset. Each module should contain as many genes as possible. We adopt the following mathematical definition of gene expression module or a bi-cluster [12]:

Definition 1 *Let S be a set of conditions, G be a set of genes, and V(S, G) be the expression matrix. Given parameters $\alpha(0 < \alpha < 1)$ and w, a gene expression module (C, D), $C \subseteq S$, $D \subseteq G$, is a subset C of samples, and a subset D of genes that satisfies the following criteria:*

1. **Size:** *the number of samples in C is at least an α-fraction of all samples, i.e. $|C| \geq \alpha|S|$*

2. **Conservation:** *every gene in D is expressed within a small range w across all samples in C, i.e. $\forall D_k \in D$, $\forall C_i, C_j \in C \Rightarrow |V(C_i, D_k) - V(C_j, D_k)| \ \leq \ w$*

3. **Maximality:** *the module contains as many genes as possible. Every gene not in D is not conserved across all samples in C, i.e.* $\forall D_k \notin D$, $\exists C_i, C_j \in C \Rightarrow |V(C_i, D_k) - V(C_j, D_k)| > w$

Given this definition, the gene expression data may contain many modules. Among all expression modules, we are interested in the one with the largest number of genes. Intuitively, as fewer samples are included in the module, more genes will satisfy the conservation criterion (Figure 1). Our algorithm starts from a subset of samples that match the size criterion, and uses Gibbs sampling to iteratively update this sample subset to maximize the number of genes in the module. We have implemented this algorithm in the GEMS (Gene Expression Module Sampler) program. The GEMS program and its C++ source code are available online at `http://genomics10.bu.edu/terrence/gems/`.

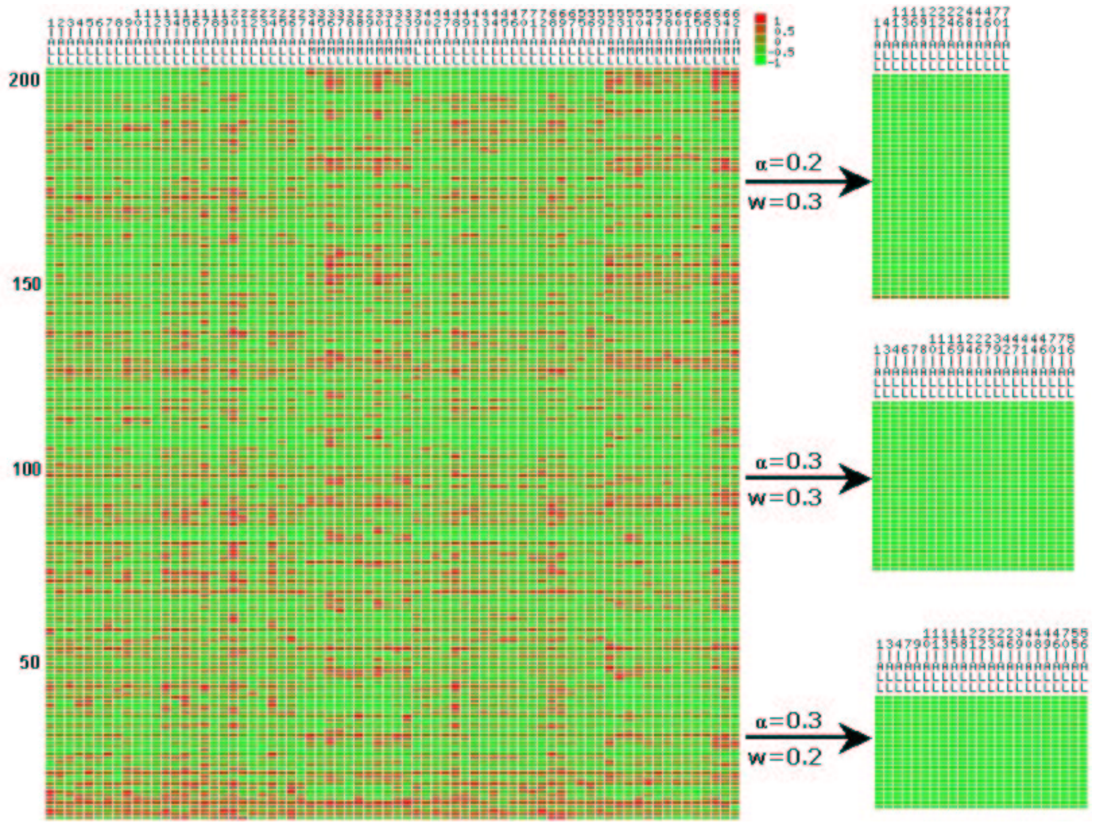

Figure 1: Expression modules extracted from a leukemia dataset [5]. As α increases, the number of genes in a module decreases. As w increases, the criterion for conservation relaxes, allowing more genes to participate in a module.

2 Methods

2.1 Description of the Algorithm (GEMS)

A gene expression module in this paper is defined by a subset of conditions and a subset of genes. A specific condition subset directly determines the maximal gene subset that satisfies the width constraint; if the difference between maximal and minimal expression values of a gene across the selected conditions is larger than the predefined w-value, then this gene cannot belong to the module, otherwise it will be included. Therefore, given a subset of conditions we can easily compute the maximum set of genes that are contained in the module defined by these conditions, thus reducing the problem of computing modules to the problem of search on subsets of conditions. Given a subset X of conditions, whose size is an α-fraction of the number of samples, our algorithm consists of two steps. In the first step we deploy a Gibbs sampler on X and in the second step we refine the solution using a local search.

We first sketch our randomized algorithm based on a Gibbs sampling methodology. The goal of our Gibbs sampler is to optimize the chosen condition subset so that the number of conserved genes across these conditions is the largest. Suppose there are d conditions in a selected subset X, $X = (\chi_1, \ldots, \chi_d)$ where each χ_i refers to a condition in S. A systematic-scan scheme is used such that the elements in X are sequentially updated from χ_1 to χ_d at each iteration [11]. Let $\overline{X}$ be the set of conditions that are not in X. At each single update step, one element χ_i is chosen and the condition assigned to it is removed from X. The sampler tries to draw a condition from $\overline{X}$ and assign it to the selected element χ_i. The sampling probability for every candidate condition is proportional to a scoring function $f(X)$ equal to the number of conserved genes if this condition is added to X. A condition contributing to larger score has higher probability to be assigned to X. As the sampling process progresses, this procedure aims to increase $f(X)$. After a sufficiently large number of iterations, the subset X should optimize $f(X)$, i.e., the number of conserved genes across all conditions in X. Although $f(X)$ is asymptotically and stochastically increasing, it is possible that $f(X)$ decreases in some iterations. This is characteristic of any stochastic process that attempts to escape from local minima. Consequently, we keep track of the largest score $f(X_{max})$ and its corresponding subset of conditions X_{max}. In our current implementation we use a heuristic stopping condition which checks if this "best record" has not increased for a predefined number of iterations.

Then we invoke a local procedure on X_{max} to further increase (if possible) the number of genes in the module. Let X' be the best condition subset detected by the Gibbs Sampler ($X' = X_{max}$), and be the subset of conditions that are not in X'. At each single update step, one element χ'_i is chosen from X'. By fixing all the other elements, we test all possible assignments to χ'_i. The score for every candidate condition is computed using the same scoring function as before. Instead of sampling conditions from a probability distribution, the candidate with largest score is assigned to χ'_i. In case of a tie, the first one is used. We then iterate over all χ'_i until there are no further changes during an entire iteration. The local search procedure terminates when X' converges. Let D be the subset of genes that satisfy the width constraint across X' (that is, the range of expression of each gene in D across the conditions in X' has width at most w). The gene expression module (X', D) is optimal in the sense that the size of D cannot be improved by a single change on X'.

The last step of our approach tries to increase the number of conditions in a module by scanning every condition not in X' to see if this condition can be recruited and added to the module without violating conservation criterion. The details of the module discovery procedure are given in Figure 2.

2.2 Statistical Significance by Permutation Test

In order to evaluate the significance of clusters produced by GEMS, we devised a permutation test. The permutation test relies on a "density" measure and roughly measures the number of genes included in a gene expression module of a given width computed by GEMS on a randomly shuffled variant of the expression dataset. After extracting a module from gene expression data, we randomly shuffled the values within each row to create a permuted variant of the dataset. This step was repeated to generate 100 shuffled datasets. GEMS was applied to these shuffled datasets with the same values of α and w to compute the average number of conserved genes found in a GEMS-cluster by chance. The frequency that GEMS extracted a module from these permuted datasets with as many conserved genes as from the real dataset is the p-value for permutation test.

2.3 Statistical Significance Using Fisher's Exact Test

To test whether GEMS can identify samples from the same class without being given the class labels, we evaluated the quality of modules extracted by GEMS in terms of the proportion of homogeneously

```
Algorithm: $(C, D) = GEMS(V(S,G)), \alpha, w)$

Input:
  $S$ = the set of all conditions = $\{S_1, S_2, \ldots, S_N\}$
  $G$ = the set of all genes = $\{G_1, G_2, \ldots, G_M\}$
  $V(S, G)$ = expression matrix that $V(s, g)$ is the expression level of gene $g$ at condition $s$.
  $\alpha$ = size criterion
  $w$ = conservation criterion
Output:
  A gene expression module $(C, D)$ from $(S, G)$ with the largest $|D|$.
Definition:
  $d$ = ceiling($\alpha N$) is the required number of conditions in module
  $X = (\chi_1, \ldots, \chi_d)$ is a subset of conditions, $X \subseteq S$.
  $f(X)$ = the number of genes that expressed within the width bound $w$ across all conditions in $X$.
  $X_{max}$ = a condition subset corresponding to the largest score ever computed.
Step1: Randomly generate an initial condition subset $X^{(0)}$.
       Choose $X^{(0)} = (\chi_1^{(0)}, \ldots, \chi_d^{(0)})$ uniformly at random as initial condition subset, $X_{max} = X^{(0)}$.
Step2: Systematic scan Gibbs sampler for every component in $X^{(t)}$
       Let $X^{(t)} = (\chi_1^{(t)}, \ldots, \chi_d^{(t)})$ for iteration $t$. At iteration $t+1$, we conduct the following steps:
       for $i = 1$ to $d$,
           for $j = 1$ to $N$,
               $\pi_j = \begin{cases} 0 & \text{if } S_j \in X_{\bar{i}} \\ f(X_{\bar{i}} \cup \{S_j\})/F & \text{if } S_j \notin X_{\bar{i}} \end{cases}$
                    where $X_{\bar{i}} = (\chi_1^{(t+1)}, \ldots, \chi_{i-1}^{(t+1)}, \chi_{i+1}^{(t)}, \ldots, \chi_d^{(t)})$
                          $F$ is a constant that makes $S\pi_j = 1$
               if $f(X_{\bar{i}} \cup \{\chi_i^{(t+1)} = S_j\}) > f(X_{max})$ then $X_{max} = X_{\bar{i}} \cup \{\chi_i^{(t+1)} = S_j\}$
           end $j$;
           Draw $\chi_i^{(t+1)}$ from $\{S_1, \ldots, S_N\}$ according to the conditional probability distribution
               $\pi = (\pi_1, \ldots, \pi_N)$.
       end $i$;
       Repeat Step 2 unless $X_{max}$ had been kept unchanged for at least L iterations.
Step3: Local search step starts from the best condition subset detected by the Gibbs Sampler
       Let $X'^{(0)} = X_{max}$
Step4: Perform local search scheme
       Let $X'^{(t)} = (\chi_1'^{(t)}, \ldots, \chi_d'^{(t)})$ for iteration $t$. At iteration $t+1$, we conduct the following steps:
       for $i = 1$ to $d$,
               Find $\chi_i'^{(t+1)}$ in $\{s \mid s \in S$ and $s \notin X'_{\bar{i}}\}$ to maximize $f(X')$
                   where $X'_{\bar{i}} = (\chi_1'^{(t+1)}, \ldots, \chi_{i-1}'^{(t+1)}, \chi_{i+1}'^{(t)}, \ldots, \chi_d'^{(t)})$
                   and $X' = (\chi_1'^{(t+1)}, \ldots, \chi_i'^{(t+1)}, \chi_{i+1}'^{(t)}, \ldots, \chi_d'^{(t)})$
       end $i$;
       Repeat Step 4 until $X'^{(t)} = X'^{(t+1)}$.
Step5: Find the subset $D$ of genes that satisfy the width constraint across $X'$.
       Greedily add every condition to $X'$ that does not violate the conservation constraint.
       Let $C$ be the expanded condition subset. Output $(C, D)$
```

Figure 2: The procedures deployed by GEMS for discovery of gene expression modules.

labeled samples, that is, the observed clusters are enriched in specific phenotypes more than we expect by chance. The statistical significance is given by a two-sided Fisher's exact test. Suppose there are n samples in the data with m belonging to a particular class C, and we a find a module containing n' samples, of which m' belong to class C. Then the probability that randomly selected n' samples will contain at least m' samples from class C is given by the tail of a hypergeometric distribution:

	Class C	Others	
In module	m'	n' - m'	n'
Unselected	m - m'	n - m - n' + m'	n - n'
	m	n - m	N

$$Pr = \sum_{i>=m'}^{min(n',m)} \frac{\binom{m}{i}\binom{n-m}{n'-i}}{\binom{n}{n'}}$$

For two-sided Fisher's exact test, the above probability should double. For example, there are 47 Acute Lymphocytic Leukemia (ALL) samples and 25 Acute Myelogenous Leukemia (AML) samples in the leukemia dataset [5]. In the case of one of the results reported in Table 1a, we identified a module with 29 ALL samples and 7 AML samples. The p-value of the two-sided Fisher's exact test is two times the exact probability of obtaining a phenotypic enrichment of 29 ALL samples or more in a given module:

$$2 \times Pr(\text{extracted ALL} = 29) = 2 \times \sum_{i>=29}^{36} \frac{\binom{47}{i}\binom{25}{36-i}}{\binom{72}{36}} = 0.013$$

For other modules the computation is similar.

3 Results

3.1 Synthetic Data

Synthetic data were created with a pseudo-module generator program (programmed in Perl) to evaluate the performance of the algorithm on noisy data. The procedure systematically embeds modules with a defined number of genes, conditions and width into a randomly generated background. The noise was generated by a uniform probability on a hyper-rectangular volume. All values in the synthetic dataset are in the range between 0 and 1. We applied GEMS on a 200x100 dataset in which a 100x40 module (width 0.3) was embedded. The summary of results is shown in Table 1b. The procedure was repeated with different parameters. All modules extracted by GEMS were statistically significant in both permutation test and Fisher's exact test. The embedded module was successfully extracted in the majority of cases. If the required modular size is larger than the size of embedded module, no modules were extracted. The significance of all modules extracted from the synthetic dataset using the permutation test was better than 0.01.

3.2 Leukemia Data

We applied GEMS to a leukemia dataset; see Golub *et al.* [5] for a detailed description of the data. This dataset consists of expression data from Affymetrix Genechips for 7,129 genes collected from 72 leukemia patients, of whom 47 were diagnosed with ALL, and 25 were AML patients. For each gene, we rescaled the expression values across conditions to range from -1 to 1 for standardization. Modules were extracted for different settings of α and w. Because small modules with few genes are not of interest, a threshold of 10 was put on the number of genes in modules. The summary of results is shown in Table 1a. The number of genes in every extracted module is statistically significant in the permutation test. The Fisher's exact probabilities for sample clustering are less than 0.01 in most of the cases, but they tended to increase as α value and width became larger.

3.3 Comparison with xMotif

We compared the performance of GEMS with another published bi-clustering algorithm: xMotif (version 2.0) [12]. The dataset used are the leukemia dataset having 7129 genes, and a reduced leukemia dataset consisting of 72 samples and 200 genes. We reduced the dimensionality (number of genes) to test the performance of the algorithms on both the full set and a smaller problem. The dimension reduction was implemented by identifying 200 genes that were selected by Rankgene [15] based on their ability to distinguish between ALL and AML samples. Reducing the number of genes simplifies the clustering problem. The performance results are illustrated in Table 2. Our main conclusion from this study is that GEMS and xMotif demonstrate comparable ability to cluster samples, i.e., group samples from one category into a single bi-cluster. Their efficiency (running time of the algorithm) appears to be similar as well.

Table 1: Modules extracted by GEMS from AML-ALL data (1a) and synthetic data (1b).

(1a) ALL-AML: 7129 genes x 72 samples.

w	α	#Genes	#ALL	#AML	P_F	Time (min)
0.1	0.1	95	8	0	0.044	2
	0.2	57	15	0	0.001	2
	0.3	33	23	0	<0.001	4
	0.4	14	31	5	<0.001	4
	0.5	12	38	3	<0.001	9
0.2	0.1	305	8	0	0.044	1
	0.2	140	15	0	0.001	2
	0.3	97	22	0	<0.001	4
	0.4	54	26	3	<0.001	7
	0.5	33	26	10	0.216	3
0.3	0.1	698	8	0	0.044	1
	0.2	260	15	0	0.001	3
	0.3	169	22	0	<0.001	2
	0.4	130	29	0	<0.001	2
	0.5	71	29	7	0.013	3
0.4	0.1	1359	8	0	0.044	3
	0.2	490	15	0	0.001	2
	0.3	285	22	0	<0.001	3
	0.4	214	29	0	<0.001	3
	0.5	124	28	8	0.047	10
0.5	0.1	2234	8	0	0.044	7
	0.2	830	15	0	0.001	4
	0.3	463	22	0	<0.001	7
	0.4	301	29	0	<0.001	4
	0.5	204	33	3	<0.001	5

(1b) Synthetic data: 200 genes x 100 samples containing one 200x100 module with width 0.3.

w	α	#Genes	#TP	#FP	P_F	Time (sec)
0.1	0.1	54	10	0	<0.001	10
	0.2	42	24	0	<0.001	16
	0.3	42	32	0	<0.001	32
	0.4	32	40	0	<0.001	44
	0.5	0	0	0		56
0.2	0.1	99	10	0	<0.001	16
	0.2	93	21	0	<0.001	13
	0.3	87	32	0	<0.001	20
	0.4	81	40	0	<0.001	25
	0.5	0	0	0		36
0.3	0.1	102	11	0	<0.001	15
	0.2	100	40	0	<0.001	15
	0.3	100	40	0	<0.001	17
	0.4	100	40	0	<0.001	20
	0.5	0	0	0		25
0.4	0.1	103	11	0	<0.001	11
	0.2	100	40	0	<0.001	17
	0.3	100	40	0	<0.001	13
	0.4	100	40	0	<0.001	16
	0.5	0	0	0		82
0.5	0.1	108	10	0	<0.001	18
	0.2	100	40	0	<0.001	10
	0.3	100	40	0	<0.001	21
	0.4	100	40	0	<0.001	45
	0.5	0	0	0		40

Keys: PF, p-values from two-sided Fisher's exact test; #TP, number of extracted samples that really came from embedded module; #FP, number of extracted samples that were not in embedded module.

Table 2: The comparison between GEMS and xMotif in leukemia datasets.

	#Genes	#Samples	ALL	AML	Running time
Leukemia data with 7129 genes					
GEMS	54	29	26	3	269 sec
xMotif	52	40	32	8	245 sec
	(Settings: $\alpha = 0.4$, $w = 0.2$)				
Leukemia data with 200 genes					
GEMS	46	29	29	0	21 sec
xMotif	54	34	34	0	19 sec
	(Settings: $\alpha = 0.4$, $w = 0.4$)				

4 Discussion

In this paper we have described a new bi-clustering algorithm based on a Gibbs sampling framework. The algorithm identifies statistically significant expression modules in microarray data. We tested the performance of the algorithm to discover phenotypically distinct cell types on the publicly available leukemia dataset. The procedure effectively discovered modules composed of samples mostly from one disease category (the class labels of each sample was not available to the algorithm).

Our definition of expression module is a subset of genes and a subset of conditions such that the expression values of each gene are conserved across the conditions. However, the results of microarray experiments are always noisy, and a clustering algorithm should tolerate some fraction of potential outliers in an expression module. Therefore, we tested whether the algorithm could recover modules embedded in noisy synthetic data generated by a random process. Our algorithm GEMS successfully extracted the embedded module with different parameter settings. The resulting clusters were shown significant using two statistical tests. In particular, for many of the clusters the application of Fisher's exact test, yielded two-sided p-values lower than 0.01. Naturally, for modules with a small number of conditions it is difficult to demonstrate statistical significance even for optimal clusters. At the high end of α or w values, the p-values consequently increased dramatically, since larger amount of noise was introduced. GEMS was also demonstrated to be relatively efficient. The program was tested on a Pentium IV 1.8GHz Linux platform. For the leukemia dataset with 7,129 genes and 72 conditions, the running time varied at different parameter settings, but never surpassed 10 minutes (Table 1a). One of the major factors affecting efficiency is the number of iterations in Gibbs sampling step. More sampling iterations will raise the likelihood to reach globally optimal results. Users are allowed to adjust the number of iterations balancing efficiency or performance as appropriate.

GEMS is also capable of detecting multiple modules. This feature is supported by masking the samples associated with the detected modules and rerunning the algorithm on the rest of the data. Additional modules can be extracted iteratively until no additional significant modules are generated from the unmasked portion of the data.

GEMS is an effective and relatively efficient procedure for the discovery of modules of co-expressed genes. The module information generated might be valuable for further analysis such as functional annotation, transcription factor binding site identification, and regulatory network analysis.

5 Acknowledgments

This work was supported in part by grant DBI-0239435 from NSF and grant 5 P20 GM 66401-3 from NIH.

References

[1] Califano, A., Stolovitzky, G., and Tu, Y., Analysis of gene expression microarrays for phenotype classification, *Proc. Intell. Syst. Mol. Biol.*, 8:75–85, 2000.

[2] Cheng, Y. and Church, G.M., Biclustering of expression data, *Proc. Intell. Syst. Mol. Biol.*, 8:93–103, 2000.

[3] Eisen, M.B., Spellman, P.T., Brown, P.O., and Botstein, D., Cluster analysis and display of genome-wide expression patterns, *Proc. Natl. Acad. Sci.*, 95(25):14863–14868, 1998.

[4] Getz, G., Levine, E., and Domany, E., Coupled two-way clustering analysis of gene microarray data, *Proc. Natl. Acad. Sci.*, 97(22):12079–12084, 2000.

[5] Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J.P., Coller, H., Loh, M.L., Downing, J.R., Caligiuri, M.A., Bloomfield, C.D., and Lander, E.S., Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring, *Science*, 286(5439):531–537, 1999.

[6] Hartigan, J.A., Direct clustering of a data matrix, *J. Amer. Statis.*, 67:123–129, 1972.

[7] Herwig, R., Poustka, A.J., Muller, C., Bull, C., Lehrach, H., and O'Brien, J., Large-scale clustering of cDNA-fingerprinting data, *Genome Res.*, 9(11):1093–1105, 1999.

[8] Kluger, Y., Basri, R., Chang, J.T., and Gerstein, M., Spectral biclustering of microarray data: Coclustering genes and conditions, *Genome Res.*, 13(4):703–716, 2003.

[9] Lawrence, C.E., Altschul, S.F., Boguski, M.S., Liu, J.S., Neuwald, A.F., and Wootton, J.C., Detecting subtle sequence signals: A Gibbs sampling strategy for multiple alignment, *Science*, 262(5131):208–214, 1993.

[10] Lazzeroni, L. and Owen, A., Plaid models for gene expression data, *Statistica Sinica*, 12(1):61–86, 2002.

[11] Liu, J.S., *Monte Carlo strategies in scientific computing*, Springer, 2001.

[12] Murali, T.M. and Kasif, S., Extracting conserved gene expression motifs from gene expression data, *Proc. Pacific Symp. Biocomputing '03*, World Scientific, 77–88, 2003.

[13] Segal, E., Shapira, M., Regev, A., Pe'er, D., Botstein, D., Koller, D., and Friedman, N., Module networks: Identifying regulatory modules and their condition-specific regulators from gene expression data, *Nat. Genet.*, 34(2):166–176, 2003.

[14] Sheng, Q., Moreau, Y., and De Moor, B., Biclustering microarray data by Gibbs sampling, *Bioinformatics*, 19(Suppl 2):196–205, 2003.

[15] Su, Y., Murali, T.M., Pavlovic, V., Schaffer, M., and Kasif, S., RankGene: Identification of diagnostic genes based on expression data, *Bioinformatics*, 19(12):1578–1579, 2003.

[16] Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., Lander, E.S., and Golub, T.R., Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation, *Proc. Natl. Acad. Sci.*, 96(6):2907–2912, 1999.

[17] Tanay, A., Sharan, R., and Shamir, R., Discovering statistically significant biclusters in gene expression data, *Bioinformatics*, 18(Suppl 1):136–144, 2002.

[18] Tibshirani, R., Hastie, T., Eisen, M., Ross, D., Botstein, D., and Brown, P., *Clustering methods for the analysis of DNA microarray data*, Tech. Rep., Department of Statistics, Stanford University, 1999.