Hindawi Publishing Corporation
EURASIP Journal on Applied Signal Processing
Volume 2006, Article ID 59809, Pages 1–12
DOI 10.1155/ASP/2006/59809

DNA Microarray Data Analysis: A Novel Biclustering Algorithm Approach

Alain B. Tchagang[1] and Ahmed H. Tewfik[2]

[1] *Department of Biomedical Engineering, Institute of Technology, University of Minnesota, 312 Church Street SE, Minneapolis, MN 55455, USA*
[2] *Department of Electrical and Computer Engineering, Institute of Technology, University of Minnesota, 200 Union Street SE, Minneapolis, MN 55455, USA*

Received 15 May 2005; Revised 5 October 2005; Accepted 1 December 2005

Biclustering algorithms refer to a distinct class of clustering algorithms that perform simultaneous row-column clustering. Biclustering problems arise in DNA microarray data analysis, collaborative filtering, market research, information retrieval, text mining, electoral trends, exchange analysis, and so forth. When dealing with DNA microarray experimental data for example, the goal of biclustering algorithms is to find submatrices, that is, subgroups of genes and subgroups of conditions, where the genes exhibit highly correlated activities for every condition. In this study, we develop novel biclustering algorithms using basic linear algebra and arithmetic tools. The proposed biclustering algorithms can be used to search for all biclusters with constant values, biclusters with constant values on rows, biclusters with constant values on columns, and biclusters with coherent values from a set of data in a timely manner and without solving any optimization problem. We also show how one of the proposed biclustering algorithms can be adapted to identify biclusters with coherent evolution. The algorithms developed in this study discover all valid biclusters of each type, while almost all previous biclustering approaches will miss some.

1. INTRODUCTION

One of the major goals of gene expression data analysis is to uncover genetic pathways, that is, chains of genetic interactions. For example, a researcher may be interested in identifying the genes that contribute to a disease. This task is difficult because subgroups of genes display similar activation patterns *only* under certain experimental conditions. Genes that are coregulated or coexpressed under a subset of conditions will behave differently under other conditions. Finding genetic pathways may therefore benefit from identifying clusters of genes that are coexpressed under subsets of conditions as opposed to all conditions.

Gene expression data is typically arranged in a data matrix, with rows corresponding to genes and columns corresponding to experimental conditions. Conditions can be different environmental conditions or different time points corresponding to one or more environmental conditions. The (n, m)th entry of the gene expression matrix represents the expression level of the gene corresponding to row n under the specific condition corresponding to column m. The numerical value of the entry is usually the logarithm of the relative amount of the mRNA of the gene under the specific condition. By simultaneously clustering the rows and columns of the gene expression matrix, one can identify candidate subsets of conditions that may be associated with cellular processes that exhibit themselves only or identify subsets of genes that potentially play a role in a given biological process. Biological analysis and experimentation could then confirm the biological significance of the candidate subsets.

Biclustering was first described in the literature by Hartigan [1]. It refers to a distinct class of clustering algorithms that perform simultaneous row-column clustering. The biclustering problems arise in microarray data analysis, collaborative filtering, market research, information retrieval, text mining, electoral trends, exchange analysis, and so forth. Cheng and Church were the first to apply biclustering to analyze DNA microarray experimental data [2]. They introduced the term biclustering to denote simultaneous row-column clustering of gene expression data. Biclustering algorithms are also known as bidimensional clustering, subspace clustering, and coclustering in other application fields. It should be clear that biclustering techniques produce local models, whereas clustering approaches compute global models. If we use a clustering algorithm on the rows of the gene expression matrix, a given gene cluster is defined using all the conditions. In contrast, a biclustering technique will assign a gene to a bicluster based on a subset of conditions.

Furthermore, when a clustering algorithm is applied to the rows of the gene expression matrix, it assigns each gene to a single cluster. Biclustering techniques on the other hand identify clusters that are not mutually exclusive or exhaustive. A gene may belong to no cluster, one or more clusters.

Cheng and Church compute the residue of each element of a submatrix of the gene expression matrix by subtracting from that element the means of all elements in its corresponding row and column and by adding a constant equal to the overall mean of all elements in the matrix. They define a bicluster to be a submatrix formed with a subset of rows and columns of the gene expression matrix with a low mean-squared residue score and used a greedy approach to find biclusters. After that, many other approaches were proposed in the literature [3–9]. For example, Tanay et al. [3] mapped expression data onto bipartite graphs and used probabilistic graph techniques to find biclusters. Getz et al. [4] devised a coupled two-way iterative clustering algorithm to identify biclusters. Lazzeroni and Owen [5] introduced the notion of a plaid model, which describes the input matrix as a linear function of variables corresponding to its biclusters. Ben-Dor et al. [6] defined a bicluster as an order-preserving submatrix, or equivalently, a group of genes whose expression levels induce some linear order across a subset of the conditions. Yang et al. [9] used tree traversal with two-way pruning of maximum coherent sets for each pair of genes and each pair of conditions, see [10] for many other approaches.

Most of these previous techniques search for one or two types of biclusters among four that have been identified in the literature [10]: biclusters with constant values, biclusters with constant values on rows or columns, biclusters with coherent values, and biclusters with coherent evolution. Most previous techniques are also greedy and will miss meaningful biclusters. Many of these pioneering approaches used a cost function to define biclusters. In many cases, the cost function will measure the square deviation from the sum of the mean value of expression levels in the entire bicluster, and the mean values of expression levels along each row and column in the bicluster.

Our objective here is to develop a biclustering algorithm that is able to discover *all* biclusters in a given data set of any type defined by the user in a timely manner. The proposed biclustering algorithm approach is different from previous ones in several ways. Firstly, the proposed approach can be used to find the exact number of all valid perfect biclusters in each type and identify all of them in a timely manner. Secondly, the proposed approach uses basic linear algebra and arithmetic tools and avoids the need for heuristic cost functions of prior approaches that can miss some pertinent biclusters. More specifically, our approach relies on the manipulation of elementary binary matrices with entries equal to "0" or "1." Finally, our approach allows the user to view biclusters under any specific experimental condition.

Observe also that our procedures will produce more biclusters than most of the other biclustering approaches since they identify *all* biclusters of a given type. As mentioned above, this reduces the probability of missing a bicluster of potentially significant biological value. On the other hand, this also increases the number of biclusters that a biologist needs to further examine. So far, we have not identified an effective criterion for ranking biclusters according to their potential biological significance.

The rest of this paper is organized as follows. After a quick description of the gene expression matrix in Section 2, we develop the proposed biclustering algorithm in Section 3. In Section 4, we show some simulation results and we compare the proposed biclustering algorithm with previous ones.

2. GENE EXPRESSION MATRIX

A DNA microarray data can be represented as an $N \times M$ matrix A whose rows represent the genes, columns represent the experimental conditions, and real-number entries a_{nm} represent the expression level of gene n under condition m as illustrated in

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1M} \\ a_{21} & a_{22} & \cdots & a_{2M} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nM} \\ \vdots & \vdots & \vdots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NM} \end{bmatrix}. \tag{1}$$

We can also partition the matrix A into rows, or into columns as illustrated by

$$\begin{aligned} A &= \begin{bmatrix} R_1 & R_2 & \cdots & R_n & \cdots & R_N \end{bmatrix}^T, \\ A &= \begin{bmatrix} C_1 & C_2 & \cdots & C_m & \cdots & C_M \end{bmatrix}. \end{aligned} \tag{2}$$

In (2),

$$\begin{aligned} R_n &= \begin{bmatrix} a_{n1} & a_{n2} & \cdots & a_{nm} & \cdots & a_{nM} \end{bmatrix}, \\ C_m &= \begin{bmatrix} a_{1m} & a_{2m} & \cdots & a_{nm} & \cdots & a_{Nm} \end{bmatrix}^T, \end{aligned} \tag{3}$$

where $1 \le n \le N$ and $1 \le m \le M$. The row vector R_n corresponds to the expression levels of the nth gene under M conditions. The column vector C_m corresponds to the expression levels of the N genes under the mth condition. From (1), we can also define two additional vectors: the row vector Conditions$(1 \times M)$ and the column vector Genes$(1 \times N)$. They are both label vectors and they are defined to keep track of every condition and gene:

$$\begin{aligned} &\text{conditions} \\ &\quad = \begin{bmatrix} \text{Condition 1} & \cdots & \text{Condition } m & \cdots & \text{Condition } M \end{bmatrix}, \\ &\text{genes} \\ &\quad = \begin{bmatrix} \text{Gene 1} & \text{Gene 2} & \text{Gene 3} & \cdots & \text{Gene } n & \cdots & \text{Gene } N \end{bmatrix}^T. \end{aligned} \tag{4}$$

3. THE PROPOSED BICLUSTERING ALGORITHM

Our proposed biclustering algorithm works as follows. After solving the problems of missing values, noise corruption using any of the known techniques, or a simple approach that

we describe below, the gene expression matrix is written as the sum of the product of each of its distinct elements with an elementary matrix. Each elementary matrix is binary, that is, its elements are either "1" or "0." By performing elementary row or the column operations on the elementary matrices, it becomes easy to identify all perfect biclusters in a timely manner.

3.1. Data conditioning

The first part of the proposed biclustering algorithm consists of performing the data conditioning due to the fact that we are not only working with noisy data, but also DNA experimental data contains missing values.

Many techniques to recover missing values have been developed in the literature, for example, [11, 12]. Since the recovery of missing values is not our main focus in this study, we have used the zero method, that is, replacing each missing value by zero.

Several techniques have been proposed in the literature, to deal with noise, including many data quantization techniques. In this study, we have used the following approach. First, we identify the number L of distinct values α_l that exist in the gene expression matrix A. We assume that the values α_l are rank-ordered according to their magnitudes, that is, $\alpha_l < \alpha_{l+1}$. Next, we redefine α_l using

$$\alpha_l = \frac{b_l + b_{l-1}}{2}, \tag{5}$$

where

$$\begin{aligned} b_l &= b_0 + le, \quad \text{with } l = 1\text{to } L, \\ e &= \frac{b_L - b_0}{L}, \\ b_0 &= \min([a_{nm}]), \\ b_L &= \max([a_{nm}]). \end{aligned} \tag{6}$$

The interval $[b_0 \;\; b_L]$ is then divided into L equal intervals:

$$[b_0 \;\; b_L] = [b_0 \; b_1[\; U \cdots U \; [b_{l-1} \;\; b_l[\; U \cdots U \; [b_{L-1} b_L]. \tag{7}$$

Finally, a new data matrix is obtained by quantizing each expression value a_{nm} using Algorithm 1. Specifically, if a_{nm} falls in the interval $[b_{l-1} \;\; b_l[$, then it is quantized to the centroid α_l of that interval.

One advantage of using this quantization approach is that it does operate on all the data of the matrix. Therefore the biclusters that are present in the original set of data are not likely to be destroyed. All it does is reducing the number of original biclusters and increasing their size by merging some of them together. This happens because this first global manipulation reduces the effect of noise in the entries of the gene expression matrix and the set of data becomes more uniform. We have also found this quantization approach to be useful in extending our basic biclustering approaches to deal with the coherent evolution case, as we will explain below.

```
Input A = microarray data
Output A = quantized microarray data
Begin,
Compute: L, b_L, b_0, e, b_l, α_l
For l = 1 to L
        For n = 1 to N
                For m = 1 to M
                        If a_nm ∈ [b_{l-1} b_l[
                                a_nm = α_l
                        elseif  a_nm == b_L
                                a_nm = α_L
                        End
                End
        End
End
End Begin
```

Algorithm 1: Data quantization procedure.

Note that one can also choose to perform the same manipulation described above gene by gene, that is, by performing the same manipulation on each row of the gene expression matrix separately. One can also use any other quantization method, such as [13].

Finally, note that it is important in practice to assess the effects of the quantization step on the biclusters that are identified by the procedures that we discuss below. This can be done by performing a simple sensitivity analysis in which the parameter e is perturbed about its selected value. It is enough to consider one or two values for e below and above its selected numerical value as determined above. Only biclusters that continue to be identified by the algorithms as e is varied should be retained for further examination. Note that the number of genes in these biclusters may also change. The user therefore needs to determine a rule for dealing with genes that may be dropped from the biclusters as e changes. The most conservative approach would be to retain only the genes that remain in the biclusters for all values of e around its selected value.

3.2. Gene expression matrix decomposition

The second part of the proposed biclustering algorithm consists of writing matrix A as the sum of the products of each of its distinct elements with a corresponding elementary matrix. It is the first important step of the proposed biclustering algorithm because after the gene expression matrix is written as mentioned above, obtaining perfect biclusters is straightforward. This is due to the fact that the elementary matrices consist of "0's" and "1's."

Given that A is made up of L distinct values, A can be expressed using

$$A = \sum_{l=1}^{l=L} \alpha_l A_l = \alpha_1 A_1 + \cdots + \alpha_L A_L. \tag{8}$$

From (8), we observe that the A_l's are binary matrices as mentioned earlier. We can also partition the matrices A_l as rows or columns as illustrated by (9) and (10), respectively:

$$A_l = \begin{bmatrix} r_1^l & r_2^l & \cdots & r_n^l & \cdots & r_N^l \end{bmatrix}^T, \tag{9}$$

$$A_l = \begin{bmatrix} c_1^l & c_2^l & \cdots & c_m^l & \cdots & c_M^l \end{bmatrix}^T. \tag{10}$$

In (9) and (10), respectively, the row vectors r_n^l are binary $1 \times M$ vectors and the column vectors c_m^l are binary $N \times 1$ vectors. The row vector r_n^l corresponds to the nth row of the elementary matrix that is associated to the lth distinct element of the gene expression matrix. The column vector c_m^l corresponds to the mth column of the elementary matrix that is associated to the lth distinct element of the gene expression matrix. From (2)–(10), we can derive the following relations:

$$\begin{aligned} R_n &= \sum_{l=1}^{l=L} \alpha_l r_n^l, \qquad C_m = \sum_{l=1}^{l=L} \alpha_l c_m^l, \qquad \sum_{l=1}^{l=L} A_l = \text{ones}(N, M), \\ &\sum_{l=1}^{l=L} r_n^l = \text{ones}(1, M), \qquad \sum_{l=1}^{l=L} c_m^l = \text{ones}(N, 1), \end{aligned} \tag{11}$$

where

$$\begin{aligned} &\alpha_1 < \alpha_2 < \alpha_3 < \leq \longleftarrow \leq \longleftarrow \leq \longleftarrow < \alpha_{l-1} \\ &< \alpha_l < \leq \longleftarrow \leq \longleftarrow \leq \longleftarrow < \alpha_{L-1} < \alpha L. \end{aligned} \tag{12}$$

Here, ones(K, L) denotes a $K \times L$ matrix of ones. Finally, note that since we are dealing with binary numbers, the number of distinct combinations that the row vector r_n^l can take is less than or equal to $2^M - 1$ and the number of distinct combinations that the column vector c_m^l can take is less than or equal to $2^N - 1$.

Decomposing the gene expression matrix as shown above has many advantages. Firstly, as mentioned earlier, all subsequent algorithms operate on binary data. Thus we gain in terms of computational complexity and memory resources. Secondly, it allows the user to get more local information about the gene expression matrix in a simple way. For example, the ones in the binary row vector r_n^l show the positions (i.e., the conditions) at which the nth gene has the same expression value α_l (which corresponds to the lth distinct element of the gene expression matrix) and its zeros show the position at which the same nth gene is not expressed at α_l. On the other hand, the ones in the binary column vector c_m^l show subgroups of genes that have the same expression value α_l (which corresponds to the lth distinct element of the gene expression matrix) under the same mth condition, and its zeros show the subgroup of genes that are not expressed at the same value α_l under the same mth condition. Also, if one is given two genes with two different binary row vectors r_n^l and r_k^l associated with the same expression value α_l, one can identify the position at which both genes are expressed simultaneously at α_l by computing the elementwise product of r_n^l and r_k^l. The result will be a binary row vector with its ones showing the positions at which both genes are expressed simultaneously at α_l. As will become clear below, this observation plays a critical role in the elaboration of the proposed biclustering algorithm. Finally, observe that the decomposition is also a powerful gene expression visualization tool.

3.3. Biclusters identification

The third part of the proposed algorithm consists of identifying the four types of biclusters from the gene expression matrix. Firstly, we develop three simple algorithms that can be used to extract all biclusters with constant values, biclusters with constant values on columns, and biclusters with constant values on rows. Secondly, we show how one of these algorithms can be modified to extract biclusters with coherent values. Finally, we describe how the modified algorithm, when coupled with tuning parameter $e (e = (b_L - b_0)/L)$ defined above, can predict biclusters with coherent evolution from a set of data.

3.3.1. Biclusters with constant values

In a DNA microarray experimental data, a perfect bicluster with constant values is any submatrix $B = [a_{ij}]$ of A with dimension $I \times J$ whose elements are constant:

$$B = [a_{ij}] = \mu \cdot \text{ones}(I, J), \tag{13}$$

where $1 \leq i \leq I$ and $1 \leq j \leq J$. Such matrices reveal subgroups of genes with constant expression levels within a subgroup of conditions or vice versa.

From the gene expression matrix decomposition performed above, such matrices can be obtained by analyzing each elementary matrix A_l separately to obtain subgroups of genes that have constant expression level α_l under different conditions. Such matrices will therefore correspond to subgroup of matrices of each elementary matrix whose elements are only the binary number "1." To identify such matrices, we proceed by identifying the set of distinct rows of each elementary matrix that are nonzeros. The sum of the cardinalities of the sets of distinct rows of each of the elementary matrices A_l will also be equivalent to the exact number of biclusters with constant values that can be found in a set of data.

In other words, since A_l is a binary matrix, and since the number of genes N is always greater than the number of conditions M, the number of biclusters (N_b) with constant values in a DNA microarray experimental data can be defined using

$$N_b = \sum_{l=1}^{l=L} P_l, \tag{14}$$

where P_l is the number of distinct nonzeros rows r_i^l of each elementary matrix A_l. Now note that each distinct nonzeros row $r_i^l$of each elementary matrix A_l constitutes the principal row element of the ith bicluster B_i^l of the elementary matrix A_l considered. Therefore, in order for any other row r_n^l of the elementary matrix A_l to belong to the ith bicluster, (15) has to be true:

$$r_i^l \cdot^* r_n^l = r_i^l, \tag{15}$$

```
Input: A = quantized microarray data
Output: B_i^l = biclusters with constant values
Begin,
Compute: P_l, r_i^l, r_n^l
    For l = 1 to L
        For i = 1 to P_l
            B_i^l = [];
            For n = 1 to N
                If r_i^l .* r_n^l == r_i^l
            B_i^l = [B_i^l; [Genes(n) α_l r_i^l]]
                End
            End
        End; B_i^l = [[0 Conditions]; B_i^l];
End Begin
```

Algorithm 2: Algorithm for finding biclusters with constant values.

```
Input: A = quantized microarray data
Output: B_j = biclusters with constant values on columns
Begin,
Compute: P_c, c_j, c_m^l
    For j = 1 to P_c
        B_j = [];
        For l = 1 to L
            For m = 1 to M
                If c_j .* c_m^l == c_j
        B_j = [B_j[Conditions(m); α_l c_j]]
                End
            End
        End; B_j = [[0 Genes]B_j];
    End
End Begin
```

Algorithm 3: Algorithm for finding biclusters with constant values on columns.

where $1 \leq i \leq P_l$, $1 \leq n \leq N$, $1 \leq l \leq L$, and "$\cdot*$" denotes the elementwise product of the two given row vectors. Algorithm 2 is then used to extract biclusters that have constant expression level α_l.

3.3.2. Biclusters with constant values on columns

A perfect bicluster with constant values on a column is any submatrix $B = [a_{ij}]$ of A with dimension $I \times J$ which has one of the following forms:

$$B = [a_{ij}] = \begin{cases} [\mu + \beta_j], & \text{additive model,} \\ [\mu\beta_j], & \text{multiplicative model.} \end{cases} \tag{16}$$

The general form can be represented using

$$B = \begin{bmatrix} \cdot & \cdot & \cdots & \cdot \\ \mu_1 & \mu_2 & \cdots & \mu_J \\ \cdot & \cdot & \cdots & \cdot \end{bmatrix}. \tag{17}$$

We observe that if $\beta_j = 0$ in the additive model or $\beta_j = 1$ in the multiplicative model, we have $a_{ij} = \mu$. Thus some perfect biclusters with constant values are also subclasses of biclusters with constant values on columns.

In a DNA microarray experimental data, biclusters with constant values on columns identify subgroups of conditions within which a subgroup of genes present similar expression values assuming that the expression values may differ from condition to condition.

Unlike Algorithm 2 which dealt with the elementary matrices A_l one at a time, identification of biclusters with constant values on columns must examine all elementary matrices at the same time. It proceeds by identifying the exact number of distinct columns of the entire elementary matrices. The number found corresponds to the exact number of biclusters with constant values on columns that can be found in a set of data. Each distinct column also defines the membership in a bicluster as shown below.

From the gene expression matrix decomposition performed above, the number of biclusters (N_b) with constant values on columns is given by

$$N_b = P_c, \tag{18}$$

where P_c is the number of distinct nonzeros columns c_j of the entire elementary matrices A_l. Once more, each distinct column c_j of the entire elementary matrices A_l constitutes the principal column element of the jth biclusters B_j. Therefore, in order for any other column c_m^l of any elementary matrix A_l to belong to the jth bicluster, (19) has to be verified:

$$c_j \cdot* c_m^l = c_j, \tag{19}$$

where $1 \leq j \leq P_c$, $1 \leq m \leq M$, and $1 \leq l \leq L$. Algorithm 3 is then used to extract biclusters that have constant values on columns.

3.3.3. Biclusters with constant values on rows

A perfect bicluster with constant values on rows is any submatrix $B = [a_{ij}]$ of A with dimension $I \times J$ which has one of the following forms:

$$B = [a_{ij}] = \begin{cases} [\mu + \alpha_i], & \text{additive model,} \\ [\mu\alpha_i], & \text{multiplicative model.} \end{cases} \tag{20}$$

The general form of such biclusters can be represented using

$$B = \begin{bmatrix} \cdots & \mu_1 & \cdots \\ \cdots & \mu_2 & \cdots \\ \cdots & \cdots & \cdots \\ \cdots & \mu_I & \cdots \end{bmatrix}. \tag{21}$$

We observe that if $\alpha_i = 0$ in the additive model or $\alpha_i = 1$ in the multiplicative model, we have $a_{ij} = \mu$. Thus perfect biclusters with constant values are subclasses of biclusters with constant values on rows.

```
Input: A = quantized microarray data
Output: B_i = biclusters with constant values on rows
Begin,
Compute: P_r, r_i, r_n^l
    For i = 1 to P_r
        B_i = [];
        For l = 1 to L
            For n = 1 to N
                If r_i .* r_n^l == r_i
            B_i = [B_i; [Genes(n)α_l r_i]]
                End
            End
        End; B_i = [[0 Conditions]; B_i];
    End
End Begin
```

Algorithm 4: Algorithm for finding biclusters with constant values on rows.

In a DNA microarray experimental data, biclusters with constant values on rows represent subgroups of genes with similar expression level across a subgroup of conditions, allowing the expression levels to differ from gene to gene.

Identification of such biclusters uses the same methodology as in Algorithm 3. Algorithm 4 operates on the rows of all the elementary matrices at the same time. It proceeds by identifying the exact number of distinct rows of the entire elementary matrices. Once more, the number found corresponds to the exact number of biclusters with constant values on rows that can be found in a set of data. Each distinct row also defines the membership in a bicluster as shown below.

From the gene expression matrix decomposition performed above, the number of biclusters (N_b) with constant values on rows is given by

$$N_b = P_r, \tag{22}$$

where P_r is the number of distinct nonzeros rows r_i of the entire elementary matrices A_l. Each distinct row r_i of the entire elementary matrices A_l constitutes the principal row element of the ith bicluster B_i. Therefore, in order for any other row r_n^l to belong to the ith bicluster, (23) has to be verified:

$$r_i \cdot^* r_n^l = r_i, \tag{23}$$

where $1 \le i \le P_r$, $1 \le n \le N$, and $1 \le l \le L$. Algorithm 4 is then used to extract biclusters that have constant value on rows.

3.3.4. Biclusters with coherent values

A perfect bicluster with coherent values is any submatrix $B = [a_{ij}]$ of A with dimension $I \times J$ which has one of the following forms:

$$B = [a_{ij}] = \begin{cases} [\mu + \alpha_i + \beta_j], & \text{additive model,} \\ [\mu\alpha_i\beta_j], & \text{multiplicative model.} \end{cases} \tag{24}$$

In this study, we will only deal with the additive model. From the above definition, we observe that the types of biclusters defined previously are particular cases of bicluster with coherent values.

(i) If $\alpha_i = \beta_j = 0$, then $a_{ij} = \mu$ and the bicluster has constant values.
(ii) If $\alpha_i = 0$, then $a_{ij} = \mu + \beta_j$ and the bicluster has constant values on columns.
(iii) If $\beta_j = 0$, then $a_{ij} = \mu + \alpha_i$ and the bicluster has constant values on rows.

In a DNA microarray experimental data, biclusters with coherent values represent subgroups of genes and subgroups of conditions with coherent values on both rows and columns.

Note that a bicluster B with coherent values can be viewed as the sum of three matrices: B_1 with constant values, B_2 with constant values on rows, and B_3 with constant values on columns, that is, $B = [\mu + \alpha_i + \beta_j] = [\mu] + [\alpha_i] + [\beta_j]$, with $B_1 = [\mu]$, $B_2 = [\alpha_i]$ and $B_3 = [\beta_j]$. Therefore, to obtain perfect biclusters with coherent values from a DNA microarray experimental data, one of the following three approaches can be used.

Approach 1

The gene expression matrix A is first written as the sum of three matrices Z_1, Z_2, and Z_3, where Z_1 is a matrix with constant values on rows, Z_2 a matrix with constant values on columns, and $Z_3 = A - (Z_1 + Z_2)$. Next, use Algorithm 2 to extract all perfect biclusters with constant values from Z_3. Finally, add to each entry of each of these biclusters the corresponding entry in $(Z_1 + Z_2)$ to obtain the biclusters with coherent values in A.

Approach 2

The gene expression matrix A is first written as the sum of three matrices Z_1, Z_2, and Z_3, where Z_1 is a matrix with constant values, Z_2 a matrix with constant values on rows, and $Z_3 = A - (Z_1 + Z_2)$. Next, use Algorithm 3 to extract all perfect biclusters with constant values on columns from Z_3. Finally, add to each entry of each of these biclusters the corresponding entry in $(Z_1 + Z_2)$ to obtain the biclusters with coherent values in A.

Approach 3

The gene expression matrix A is first written as the sum of three matrices Z_1, Z_2, and Z_3, where Z_1 is a matrix with constant values, Z_2 a matrix with constant values on columns, and $Z_3 = A - (Z_1 + Z_2)$. Next, use Algorithm 4 to extract all perfect biclusters with constant values on rows from Z_3. Finally, add to each entry of each of these biclusters the corresponding entry in $(Z_1 + Z_2)$ to obtain the biclusters with coherent values in A.

In this study, we use the third approach. The choice of the matrix $Z_1 + Z_2$ which has constant values on columns

is not arbitrary. It must be constructed using each row of the gene expression matrix A that is also part of the bicluster with coherent values as explained below.

Property 1. Let X be a matrix that contains a bicluster with coherent values embedded within its structure. Subtract from X a matrix Y that has constant values on columns, and is constructed using a row of X that is also part of the bicluster with coherent values. The resulting matrix Z contains a bicluster with constant values on rows embedded within its structure. Furthermore, the location of the bicluster with constant values in Z corresponds to that of the bicluster with coherent values in A.

Proof. Without loss of generality, consider a matrix X that includes a bicluster with coherent values embedded in it:

$$X = \begin{bmatrix} a & \alpha_1+\beta_2 & f & \alpha_1+\beta_4 & \alpha_1+\beta_5 \\ b & e & g & j & k \\ c & \alpha_3+\beta_2 & h & \alpha_3+\beta_4 & \alpha_3+\beta_5 \\ d & \alpha_4+\beta_2 & i & \alpha_4+\beta_4 & \alpha_4+\beta_5 \end{bmatrix}. \tag{25}$$

The bicluster with coherent values $B = (\alpha_i + \beta_j)$ embedded within the structure of X is

$$B = \begin{bmatrix} \cdot\cdot & \alpha_1+\beta_2 & \cdot\cdot & \alpha_1+\beta_4 & \alpha_1+\beta_5 \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & \alpha_3+\beta_2 & \cdot\cdot & \alpha_3+\beta_4 & \alpha_3+\beta_5 \\ \cdot\cdot & \alpha_4+\beta_2 & \cdot\cdot & \alpha_4+\beta_4 & \alpha_4+\beta_5 \end{bmatrix}. \tag{26}$$

Thus we can construct the matrix Y that has constant values on columns using either the first, the third, or the fourth row of X. Let us use the first row of X. Therefore, we have

$$Y = \begin{bmatrix} a & \alpha_1+\beta_2 & f & \alpha_1+\beta_4 & \alpha_1+\beta_5 \\ a & \alpha_1+\beta_2 & f & \alpha_1+\beta_4 & \alpha_1+\beta_5 \\ a & \alpha_1+\beta_2 & f & \alpha_1+\beta_4 & \alpha_1+\beta_5 \\ a & \alpha_1+\beta_2 & f & \alpha_1+\beta_4 & \alpha_1+\beta_5 \end{bmatrix}. \tag{27}$$

By computing $Z = X - Y$, we have

$$Z = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ b-a & e-\alpha_1-\beta_2 & g-f & j-\alpha_1-\beta_4 & k-\alpha_1-\beta_5 \\ c-a & \alpha_3-\alpha_1 & h-f & \alpha_3-\alpha_1 & \alpha_3-\alpha_1 \\ d-a & \alpha_4-\alpha_1 & i-f & \alpha_4-\alpha_1 & \alpha_4-\alpha_1 \end{bmatrix}. \tag{28}$$

Observe that Z has a bicluster Bc with constant values on rows embedded within its structure. Furthermore, the location of Bc corresponds to that of the bicluster with coherent values in X:

$$Bc = \begin{bmatrix} \cdot\cdot & 0 & \cdot\cdot & 0 & 0 \\ \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & \alpha_3-\alpha_1 & \cdot\cdot & \alpha_3-\alpha_1 & \alpha_3-\alpha_1 \\ \cdot\cdot & \alpha_4-\alpha_1 & \cdot\cdot & \alpha_4-\alpha_1 & \alpha_4-\alpha_1 \end{bmatrix}. \tag{29}$$

In [14], we provide a development of all of the other approaches. □

Since we do not have any knowledge about the rows of the gene expression matrix A, the intuitive approach is to use an iterative multistep approach. Specifically, we iteratively construct the matrix $Z_1 + Z_2$ with constant values on columns using each row of A. After each iteration, we compute $Z_3 = A - (Z_1 + Z_2)$ and use Algorithm 4 to extract all perfect biclusters with constant values on rows from Z_3. Finally, we add to each entry of each of these biclusters the corresponding entry in $(Z_1 + Z_2)$ to obtain the biclusters with coherent values in A.

From the proof of the above property, we observe that there are many ways to construct the matrix Z_1+Z_2 with constant values on columns and obtain the same bicluster with coherent values. Therefore, to avoid redundancy and gain in computational time, we need a strategy that prevents the algorithm from identifying a bicluster more than once. The strategy should take into account the fact that a row of the gene expression matrix can be part of more than one bicluster with coherent values. Such strategy is still under investigation.

3.3.5. Biclusters with coherent evolution

The last type of biclusters addressed in this study is the set of biclusters that exhibit coherent evolution. Identifying such biclusters can be helpful in the sense that in some applications, one might be interested in looking for subgroups of genes that are upregulated or downregulated across a subgroup of conditions without taking into account their actual expression values.

To extract such biclusters from a DNA microarray experimental data, we use the following approach. First, we tune parameter $e(e = (b_L - b_0)/L)$ defined in Section 3.1. Second, we use the definition of perfect biclusters with coherent values to obtain biclusters with coherent values from the new set of data. The location of the perfect biclusters obtained from the new set of data corresponds to that of potential biclusters with coherent evolution in the original set of data. Finally, we use a merit function to validate all resulting potential biclusters as we explain below.

By tuning parameter e defined in Section 3.1, we decrease the number L of distinct values contained in the original set of data. Thus the resulting new set of data is more uniform than the original one. By applying the algorithm that extracts biclusters with coherent values to the new set of data, we obtain perfect biclusters with coherent values. A few examples are shown and discussed below in Section 4.2. After tuning, extraction, and matching of the set of perfect biclusters obtained from the new set of data with their equivalent in the original set of data, we obtain subgroups of genes with expression levels that evolve coherently or stay constant across a subgroup of conditions regardless of their expression values. In some cases, we get biclusters with 1 or 2 imperfections. By imperfection we mean a gene with expression levels that do not evolve coherently with those of all other genes for a few conditions.

In this study, we have used the same merit function as previous researchers [10] to validate potential biclusters with

coherent evolution. Specifically, we adopt the mean-squared residue function H defined by

$$H(I,J) = \frac{1}{|I||J|} \sum_{i\in I, j\in J} r(a_{ij})^2. \tag{30}$$

In (30), $r(a_{ij}) = a_{ij} - a_{iJ} - a_{Ij} + a_{IJ}$ is the residue function,

$$a_{iJ} = \frac{1}{|J|} \sum_{j\in J} a_{ij} \tag{31}$$

is the mean of the ith row in the bicluster,

$$a_{Ij} = \frac{1}{|I|} \sum_{i\in I} a_{ij} \tag{32}$$

is the mean of the jth column in the bicluster, and

$$a_{IJ} = \frac{1}{|I||J|} \sum_{i\in I, j\in J} a_{ij} \tag{33}$$

is the mean of all the elements of the bicluster.

The residue of perfect biclusters is zero, so is their mean-squared residue. In order to validate a bicluster, we define a threshold δ and all qualified biclusters must verify:

$$H(I,J) < \delta. \tag{34}$$

3.3.6. Complexity analysis

We can easily estimate the complexity of the proposed approach. Recall that N is the number of rows of the gene expression matrix A, M is the number of columns in A, and L is the number of distinct values in A.

Algorithm 1, which is used for data quantization, requires about $(N \times M \times L)$ operations. One has to note that this step is optional. After data quantization, we perform the matrix decomposition that requires about $(N \times M \times L)$ operations. Algorithm 2 which is used to extract biclusters with constant values uses $O((N\times M+N+K+K\times M)\times L\times N_b)$ operations because we perform $N \times M$ binary multiplications, N comparisons, and K assignments $L \times N_b$ times. Here, N_b is the number of biclusters and K is the number of times (15) is verified. It can be similarly verified that the complexities of Algorithms 3 and 4 are, respectively, $O((N \times M + M + K_1 + K_1\times N)\times L\times N_b)$ and $O((N\times M+N+K_2+K_2\times M)\times L\times N_b)$, where K_1 and K_2 are the number of times (19) and (23) are verified.

From the above observations, the complete biclustering approach has complexity of $O(N \times M \times L \times N_b)$. Therefore, The proposed biclustering algorithm is less complex than the FLOC algorithm proposed by Yang et al. which has complexity $O((N + M)^2 \times K \times P)$, where P is the desired number of biclusters and K is the number of iteration till the end. FLOC was shown by Yang et al. to be less complex than the Cheng-Church algorithm [9].

4. RESULTS

Let us conclude by discussing some of the results that we have obtained. As in [13], we have implemented the proposed biclustering algorithm in Matlab and tested it on the yeast gene microarray data that can be found at [15]. The data consists of 2884 genes and 17 conditions. We have obtained the following first results. Initially, the data contained $L = 206$ distinct values.

4.1. First set of results

In the first set of results that we report here, we set $b_L = \max[a_{nm}] = 595$, $b_0 = \min[a_{nm}] = 0$, thus $e = 2.8883$ and $b_l = b_0 + le = 2.8883l$, with $1 \le l \le L$. After data conditioning, we obtained $L = 111$ new distinct values. Then from our simulation, we obtained $N_b = 10225$ biclusters with constant values, $N_b = 3391$ biclusters with constant values on rows, and $N_b = 836$ biclusters with constant values on columns. Because of the large number of biclusters found, we will present here a few illustrative results that will help the reader to grasp the magnitude of the problem and the nature of the results produced by the algorithm. Figure 1 shows an example of perfect biclusters with constant values, perfect biclusters with constant values on rows, and perfect biclusters with constant values on columns obtained. Figure 2 shows an example of perfect biclusters with coherent values obtained.

4.2. Second set of results

In the second set of results that we report, we explore the effect of two parameters: parameter e that defines the number of distinct values of the data set and threshold δ that qualifies the biclusters obtained.

For the threshold δ, we simply compare the residue of the biclusters obtained with the average residue of the Cheng-Church algorithm (204.293), and the average residue of the biclustering algorithm defined by Yang et al. (187.543) [9].

To explore the effect of e, we successively tuned its value from 2.8883 as initially defined to about 40. It is obvious that by increasing the value of e, the size of the biclusters obtained will increase and the probability of having the biclusters affected by imperfection will also increase. Figure 3 shows an example of biclusters with coherent evolution obtained without any imperfection. Thus, there is no need to use the merit function for validation. Figure 4 shows an example of perfect biclusters with coherent values obtained in the new data set after e is tuned up. Figure 5 shows the equivalent bicluster with the original data set. We observe a few imperfections, and thus need to use the merit function for validation.

For comparison, we select $\delta = 186.543$, a value that corresponds to the average value chosen by Yang et al. [9], and we set $e = 25$. In [9], Yang et al. identified 100 biclusters with an average of 195 genes and 12.8 conditions. In contrast, our procedure identified 258 biclusters with an average of 204 genes and 13 conditions or more. On the other hand, Cheng and Church identified 100 biclusters with an average of 167 genes and 12 conditions and an average value of $\delta = 204.294$. Clearly, our algorithm identifies more biclusters for the same

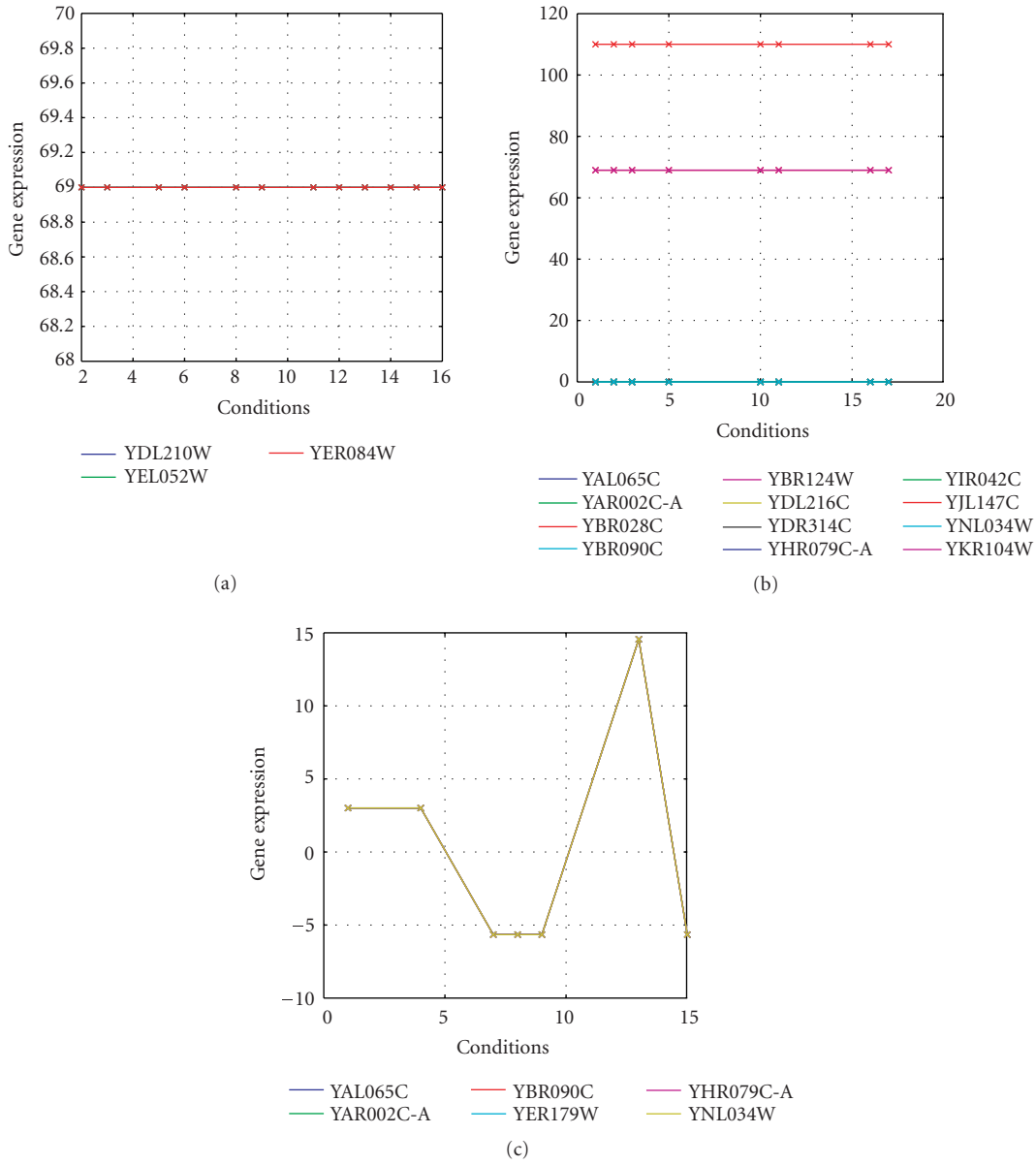


Figure 1: Example of bicluster (a) with constant values; (b) with constant values on rows; and (c) with constant values on columns.

threshold value δ. We discuss the biological significance of the biclusters that the procedure identified in the next subsection.

Note that the data conditioning and decomposition steps of our procedure took approximately 250 seconds to process the yeast data found at [15]. It took less than 10 seconds to identify a bicluster. Thus its running time is better than that of [2], which reportedly takes 300–400 seconds to find a single bicluster, and is comparable to that of [16].

4.3. Biological significance

Since our ultimate goal is to be able to uncover genetic pathways from the set of biclusters that our methods produce, we need to investigate the biological significance of these biclusters. Ideally, the investigation would also yield a criterion for ranking biclusters according to their biological significance. As mentioned earlier, we have not succeeded so far in identifying such a criterion. We will therefore limit ourselves in this subsection to a discussion of the biological significance of the 258 biclusters mentioned in Section 4.2. The analysis of these biclusters is representative of what we have seen so far. It also illustrates the complexity of the additional investigations that must be performed on the biclusters once they have been identified.

A preliminary assessment of the biological significance of the biclusters is currently under investigation using the functional categories from the Comprehensive Yeast Genome Database (CYGD) [17, 18]. The CYGD database categorizes yeast genes into fine groupings using an annotation system

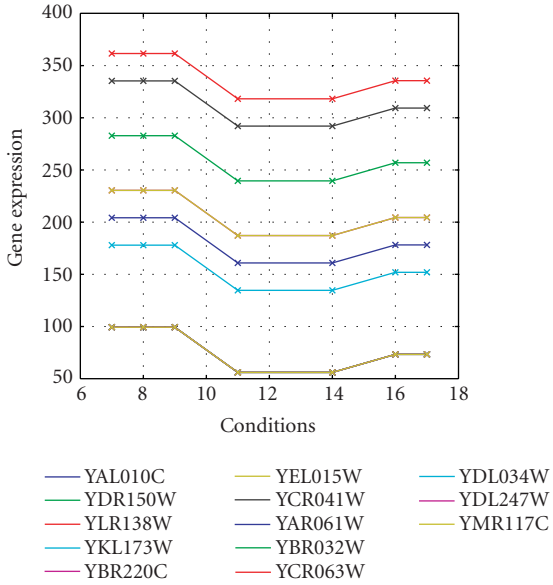


Figure 2: Example of bicluster with coherent values.

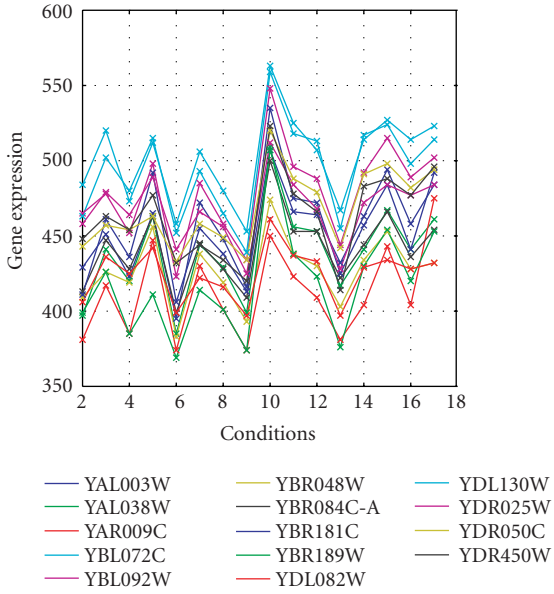


Figure 3: Example of bicluster with coherent evolutions obtained from the new data set after e is tuned up.

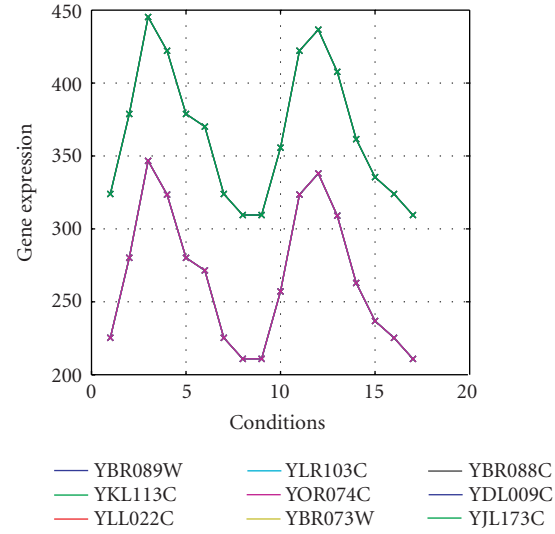


Figure 4: Example of perfect biclusters with coherent values obtained from the new data set after e is tuned up.

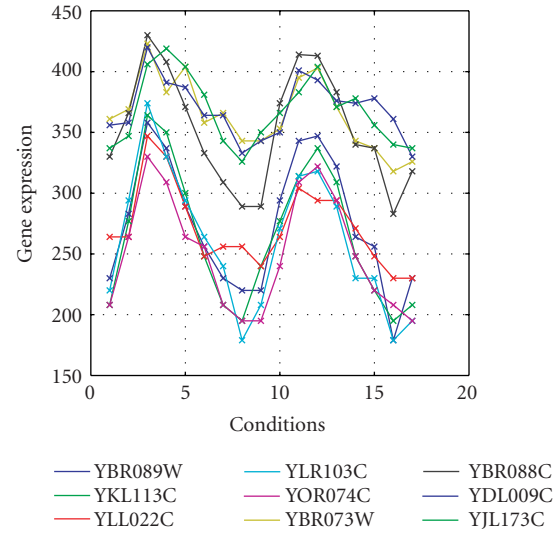


Figure 5: Equivalent of the perfect biclusters with coherent values shown in Figure 4 in the real data set with few imperfection. The lines represent different genes.

called *FunCat*, the functional classification catalog. More information can be found in [19].

Table 1 provides a preliminary biological significance analysis of the 258 biclusters in Section 4.2. The second row of Table 1 lists how many biclusters were found. Rows three through five show how many biclusters belong to one of 4 mutually exclusive categories. The third row shows how many of those biclusters contained genes that were all annotated under the same function. An example of a bicluster in this grouping would be three genes that all produce proteins whose main purpose is metabolism. The fourth row displays how many of the biclusters picked up only genes that were unclassified. The fifth row lists the number of biclusters that contained genes annotated to the same function as well as unclassified genes.

Interestingly, the algorithm picks up biclusters that are completely comprised of functionally unclassified genes. Another unexpected result is that the algorithm is able to pick up biclusters that contained "mixed" data. Another unexpected result was the number of biclusters that contained

Table 1: Biological analysis of the 258 biclusters with coherent evolutions.

Number of conditions	13	14	15	16	17
Number of biclusters with coherent values	148	69	35	5	1
Number of functionally defined biclusters	3 (2.0%)	1 (1.4%)	0	0	0
Biclusters composed entirely of unclassified genes	35 (23.6%)	12 (17.4%)	16 (45.8%)	0	1
Biclusters with unclassified genes and genes of one function	50 (33.8%)	37 (53.6%)	13 (37.1%)	4 (80%)	0
Biclusters with genes of mixed annotation	60 (40.6%)	19 (27.6%)	6 (17.1%)	1 (20%)	0

"mixed" data. The appearance of such biclusters led us to pose several questions that we are attempting to answer in collaboration with researchers in the biological sciences. The genes in these mixed biclusters showed patterns of coherent evolution but did not fall necessarily in the same functional category.

The presence of these biclusters may be indicative of the fact that coregulated genes do not necessarily belong to the same functional category. On the other hand, it may indicate that these genes have other unknown functions or functions that were not captured in the annotation we used. It is also possible that the expression levels of certain genes that belong to a given functional category affect those of some other genes that belong to a different functional category.

Many of the mixed biclusters are of biological interest because they contain genes that either belong to a single functional category or are unclassified. Current investigations are attempting to determine whether the unclassified genes in these biclusters do actually belong to the same functional category as the others. With colleagues, we are examining the literature to identify the theorized functions of many of the unclassified genes that appear in mixed biclusters or biclusters with unclassified genes. We are also studying alternative gene annotation sources, such as GO-slim [20], to answer some of the questions that we posed here.

5. CONCLUSION

In this study, we developed an efficient biclustering algorithm that can be used to extract from a set of data biclusters with constant values, constant values on rows, constant values on columns, and coherent values. We also described an approach for finding biclusters with coherent evolutions, this approach combines the algorithm that finds biclusters with coherent values with adaptive gene expression level quantization procedure. Since completing this work, we have also developed an alternative fast and direct approach for finding all biclusters with coherent evolutions [21] with no imperfection. In contrast to prior work, our procedure is able to find all biclusters with constant values, constant values on rows, constant values on columns, and coherent values. Furthermore, it has similar or lower complexity than that of prior work.

REFERENCES

[1] J. A. Hartigan, "Direct clustering of a data matrix," *Journal of the American Statistical Association*, vol. 67, no. 337, pp. 123–129, 1972.

[2] Y. Cheng and G. M. Church, "Biclustering of expression data," in *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology (ISMB '00)*, pp. 93–103, La Jolla, Calif, USA, August 2000.

[3] A. Tanay, R. Sharan, and R. Shamir, "Discovering statistically significant biclusters in gene expression data," *Bioinformatics*, vol. 18, supplement 1, pp. S136–S144, 2002.

[4] G. Getz, E. Levine, and E. Domany, "Coupled two-way clustering analysis of gene microarray data," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 97, no. 22, pp. 12079–12084, 2000.

[5] L. Lazzeroni and A. Owen, "Plaid models for gene expression data," *Statistica Sinica*, vol. 12, no. 1, pp. 61–86, 2002.

[6] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini, "Discovering local structure in gene expression data: the order-preserving submatrix problem," in *Proceedings of the 6th Annual International Conference on Computational Biology (RECOMB '02)*, pp. 49–57, Washington, DC, USA, April 2002.

[7] R. Sharan, A. Maron-Katz, and R. Shamir, "CLICK and EXPANDER: a system for clustering and visualizing gene expression data," *Bioinformatics*, vol. 19, no. 14, pp. 1787–1799, 2003.

[8] Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein, "Spectral biclustering of microarray data: coclustering genes and conditions," *Genome Research*, vol. 13, no. 4, pp. 703–716, 2003.

[9] J. Yang, H. Wang, W. Wang, and P. S. Yu, "Enhanced biclustering on expression data," in *Proceedings of 3rd IEEE Symposium on Bioinformatics and Bioengineering (BIBE '03)*, pp. 321–327, Bethesda, Md, USA, March 2003.

[10] S. C. Madeira and A. L. Oliveira, "Biclustering algorithms for biological data analysis: a survey," *IEEE Transactions on Computational Biology and Bioinformatics*, vol. 1, no. 1, pp. 24–45, 2004.

[11] O. Alter, P. O. Brown, and D. Botstein, "Processing and modeling genome-wide expression data using singular value decomposition," in *Microarrays: Optical Technologies and Informatics*, vol. 4266 of *Proceedings of SPIE*, pp. 171–186, San Jose, Calif, USA, January 2001.

[12] O. Troyanskaya, M. Cantor, G. Sherlock, et al., "Missing value estimation methods for DNA microarrays," *Bioinformatics*, vol. 17, no. 6, pp. 520–525, 2001.

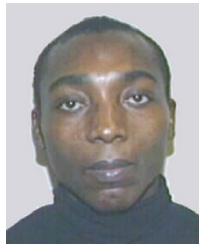
[13] A. H. Tewfik and A. B. Tchagang, "Biclustering of DNA microarray data with early pruning," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '05)*, Philadelphia, Pa, USA, March 2005.

[14] A. B. Tchagang and A. H. Tewfik, "Robust biclustering algorithm: ROBA," Tech. Rep., University of Minnesota, 2005.

[15] S. Tavazoie, J. Hughes, M. Campbell, R. Cho, and G. Church, Yeast micro data set, http://arep.med.harvard.edu/biclustering.

[16] H. Wang, W. Wang, J. Yang, and P. S. Yu, "Clustering by pattern similarity in large data sets," in *Proceedings of the International Conference on Management of Data (ACM SIGMOD '02)*, pp. 394–405, Madison, Wis, USA, June 2002.

[17] U. Güldener, M. Münsterkötter, G. Kastenmüller, et al., "CYGD: the comprehensive yeast genome database," *Nucleic Acids Research*, vol. 33, Database issue, pp. D364–D368, 2005.
[18] Munich Information Center for Protein Sequences (MIPS) and GSF-National Research Center for Environment and Health, "Comprehensive Yeast Genome Database," 2002. (visited July 21, 2005), http://mips.gsf.de/genre/proj/yeast/.
[19] A. Ruepp, A. Zollner, D. Maier, et al., "The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes," *Nucleic Acids Research*, vol. 32, no. 18, pp. 5539–5545, 2004.
[20] R. Balakrishnan, K. R. Christie, M. C. Costanzo, et al., "Saccharomyces Genome Database," http://www.yeastgenome.org.
[21] A. H. Tewfik, A. B. Tchagang, and L. Vertatschitsch, "Parallel identification of gene biclusters with coherent evolution," to appear in *IEEE Transactions on Signal Processing*, Special issue on Genomics Signal Processing.



Alain B. Tchagang received the B.S. degree and the M.S. degree in physics from the University of Yaoundé I, Cameroon, in 1996 and 1997, a "Diplome d'Ingenieur de Conception de Genie Electrique" from the "École Nationale Superieure Polytechnique" of Cameroon in 2000, the M.S. degree in electrical engineering from the University of Minnesota, USA, in October 2004. He is currently a Ph.D. student in the Department of Biomedical Engineering at the University of Minnesota. He is also a Research Assistant in the Multiscale Multirate Signal Processing Lab at the University of Minnesota. His research interests include (A) application of digital signal processing and digital control systems design to biomedical engineering (bioelectricity, biomechanics, biological transport processes, and medical imaging; (B) mathematical modeling and analysis of biological systems and data (genomics, proteomics, DNA microarray, gene expression, gene regulatory networks, and computational biology.) He did work as an Electrical Engineer Intern at http://www.cenco.us during Spring 2004, Summer 2004, Fall 2004.

Ahmed H. Tewfik received his B.S. degree from Cairo University, Cairo, Egypt, in 1982, and his M.S., E.E., and Sc.D. degrees from the Massachusetts Institute of Technology, Cambridge, Mass, in 1984, 1985, and 1987, respectively. He is the E. F. Johnson Professor of electronic communications with the Department of Electrical Engineering at the University of Minnesota. His current research interests are in genomics and proteomics, programmable wireless networks, brain computing interfaces, healthcare safety, and data-nomic and pervasive computing and storage. He is a Fellow of the IEEE. He was awarded the E. F. Johnson Professorship of Electronic Communications in 1993, a Taylor Faculty Development Award from the Taylor Foundation in 1992, and an NSF Research Initiation Award in 1990. He was selected to be the first Editor-in-Chief of the IEEE Signal Processing Letters from 1993 to 1999. He is a past Associate Editor of the IEEE Transactions on Signal Processing, was a Guest Editor of three special issues of that journal. He is currently an Associate Editor of the EURASIP Journal on Bioinformatics and Systems Biology. He also served as the President of the Minnesota chapters of the IEEE Signal Processing and Communications Societies from 2002 to 2005.