CHAPTER 1

A SURVEY ON BICLUSTERING OF GENE EXPRESSION DATA

Adelaide Freitas[1], **Wassim Ayadi**[2,4], **Mourad Elloumi**[2,5], **José Luís Oliveira**[3], **Jin-Kao Hao**[4]
[1]DMat/CIDMA, University of Aveiro, Portugal
[2]Laboratory of Technologies of Information and Communication and Electrical Engineering (LaTICE)
[3]DETI/IEETA, University of Aveiro, Portugal
[4]LERIA, University of Angers, Angers, France
[5]University of Tunis-El Manar, Tunisia.

1.1 INTRODUCTION

Microarrays allow measuring the expression level of a large number of genes under different experimental samples or environmental conditions. The data generated from them are called *gene expression data*. The extraction of biological relevant knowledge from this data is not a trivial task. Gene expression data are usually represented by a matrix M (see Table 1.1), where the i^{th} row represents the i^{th} gene, the j^{th} column represents the j^{th} condition and the cell m_{ij} represents the expression level of the i^{th} gene under the j^{th} condition.

Table 1.1 Gene expression data matrix.

	$condition_1$	...	$condition_j$	...	$condition_m$
$Gene_1$	m_{11}	...	m_{1j}	...	m_{1m}
...	...	...	...	...	...
$Gene_i$	m_{i1}	...	m_{ij}	...	m_{im}
...	...	...	...	...	...
$Gene_n$	m_{n1}	...	m_{nj}	...	m_{nm}

There are several objectives when analyzing gene expression data such as grouping subsets of genes that are coexpressed under subsets of conditions or classifying new genes, given the expression of other genes with known classification. Discovering such coexpressions can be helpful to uncover genomic knowledge such as gene networks or gene interactions. That is why, it is of utmost importance to make a simultaneous clustering of rows (genes) and columns (conditions) of the data matrix to identify clusters of genes that are coexpressed under clusters of conditions. This type of clustering is called *biclustering* [14]. The resulting clusters are called *biclusters*. A *bicluster* is a subset of genes showing similar behavior under a subset of conditions of the original expression data matrix. Let us note that a gene/condition can belong to more than one bicluster.

Formally, a *bicluster* can be defined as follows: Let $I=\{1, 2, \ldots, n\}$ be a set of indices of n genes, $J=\{1, 2, \ldots, m\}$ be a set of indices of m conditions and $M(I, J)$ be a data matrix associated with I and J. A *bicluster* associated with the data matrix $M(I, J)$ is a couple (I', J') such that $I' \subseteq I$ and $J' \subseteq J$.

The *biclustering problem* can be formulated as follows: Given a data matrix M, construct a group of biclusters B_{opt} associated with M such that:

$$f(B_{opt}) = \max_{B \in BC(M)} f(B) \tag{1.1}$$

where f is an objective function measuring the *quality*, i.e., degree of coherence, of a group of biclusters and $BC(M)$ is the set of all the possible groups of biclusters associated with M.

Clearly, biclustering is a highly combinatorial problem with a search space size $O(2^{|I|+|J|})$. In its general case, biclustering is NP-hard [14, 30].

In this chapter, we make a survey on biclustering of gene expression data. The rest of the chapter is organized as follows: First, we present the different types of biclusters and groups of biclusters. Then, we present *evaluation* *functions* and *systematic* and *stochastic* biclustering algorithms. Next, we discuss biclusters validation. Finally, we present our conclusion.

1.2 TYPES OF BICLUSTERS

A bicluster can be in one of the following cases:

1. Bicluster with constant values: It is a bicluster where all the values are equal to a constant c:
$$m_{ij} = c \tag{1.2}$$
2. Bicluster with constant values on rows or columns:
   - Bicluster with constant values on rows: It is a bicluster where all the values can be obtained by using one of the following equations:
   $$m_{ij} = c + a_i \tag{1.3}$$
   $$m_{ij} = c * a_i \tag{1.4}$$
   where c is a constant and a_i is the adjustment for the row i, $1 \leq i \leq n$.
   - Bicluster with constant values on columns: It is a bicluster where all the values can be obtained by using one of the following equations:
   $$m_{ij} = c + b_j \tag{1.5}$$
   $$m_{ij} = c * b_j \tag{1.6}$$
   where c is a constant and b_j is the adjustment for the column j, $1 \leq j \leq m$.
3. Bicluster with coherent values: It is a bicluster that can be obtained by using one of the following equations:
$$m_{ij} = c + a_i + b_j \tag{1.7}$$
$$m_{ij} = c * a_i * b_j \tag{1.8}$$
4. Bicluster with linear coherent values: It is a bicluster where all the values can be obtained by using the following equation:
$$m_{ij} = c * a_i + b_j \tag{1.9}$$
5. Or, bicluster with coherent evolution: It is a bicluster where all the rows (resp. columns) induce a linear order across a subset of columns (resp. rows).

1.3 GROUPS OF BICLUSTERS

A group of biclusters can be in one of the following cases [30]:

1. Single bicluster (Figure 1.1 (a)),
2. Exclusive rows and columns group of biclusters (Figure 1.1 (b)),
3. Non-overlapping group of biclusters with checkerboard structure (Figure 1.1 (c)),
4. Exclusive rows group of biclusters (Figure 1.1 (d)),
5. Exclusive columns group of biclusters (Figure 1.1 (e)),
6. Non-overlapping group of biclusters with tree structure (Figure 1.1 (f)),
7. Non-overlapping non-exclusive group of biclusters (Figure 1.1 (g)),
8. Overlapping group of biclusters with hierarchical structure (Figure 1.1 (h)),
9. Or, arbitrarily positioned overlapping group of biclusters (Figure 1.1 (i)).

A natural way to visualize a group of biclusters consists in assigning a different color to each bicluster and in reordering the rows and the columns of the data matrix so that we obtain a data matrix with colored blocks, where each block represents a bicluster[1].

1.4 EVALUATION FUNCTIONS

An *evaluation function* is an indicator of the performance of a biclustering algorithm. We distinguish two main classes of evaluation functions: *intra-biclusters* evaluation functions and *inter-biclusters* ones. While the former are used to quantify the coherence degree within each bicluster, the last are measures of match scores between two groups of biclusters provided by different biclustering strategies and are used to assess the ability of an algorithm to recover biclusters detected by another one.

There are several intra-biclusters evaluation functions [3, 4, 13, 14, 20, 25, 44]. Most of these functions are particular cases of the *AVerage Similarity Score* (AVSS) introduced in [29]. Given a bicluster (I', J'), the AVSS $E_{AVSS}(I', J')$ is defined as follows:

$$E_{AVSS}(I', J') = \frac{\sum_{i \in I'} \sum_{j \in J'} s_{ij}}{|I'||J'|} \tag{1.10}$$

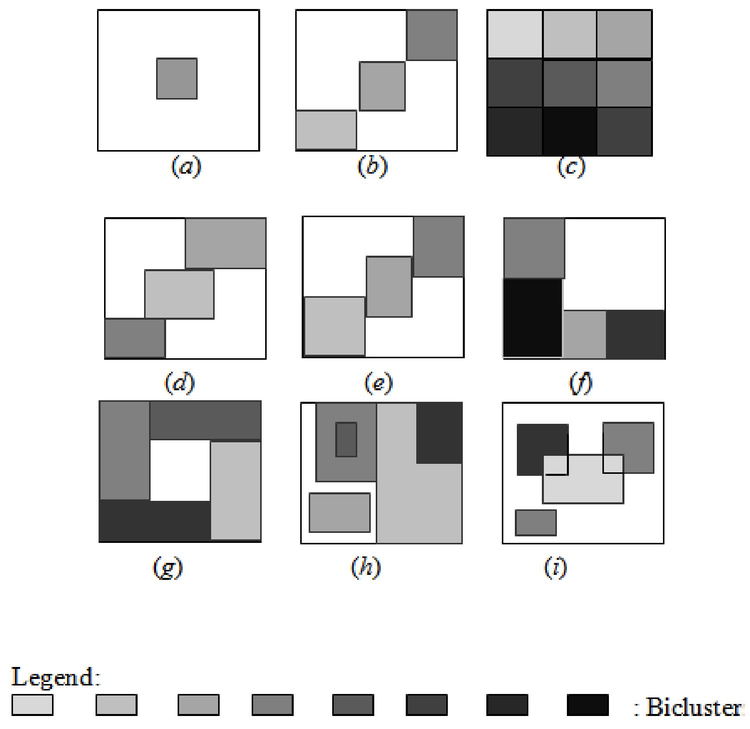

Figure 1.1 Possible structures of a group of biclusters in a data matrix.

for some similarity measure s_{ij} among elements of the row i and the column j with others elements belonging to I' and J'.

The most popular intra-biclusters evaluation function is the *Mean Squared Residue* (MSR) function, E_{MSR}, proposed in [14] (Equation 1.11). It is used by several algorithms like [3, 11, 13, 18, 33, 46, 49]. E_{MSR} is given by

$$E_{MSR}(I',J') = \frac{\sum_{i\in I'}\sum_{j\in J'}(m_{ij} - m_{iJ'} - m_{I'j} + m_{I'J'})^2}{|I'||J'|} \tag{1.11}$$

where:
$m_{I'J'}$ is the average over the whole bicluster,
$m_{I'j}$ is the average over the column j,
$m_{iJ'}$ is the average over the row i.
Since

$$\begin{aligned}&\sum_{i\in I'}\sum_{j\in J'}(m_{ij} - m_{iJ'} - m_{I'j} + m_{I'J'})^2\\ =&\sum_{i\in I'}\sum_{j\in J'}(m_{ij} - m_{I'J'})^2 - \sum_{i\in I'}\sum_{j\in J'}(m_{iJ'} - m_{I'J'})^2 - \sum_{i\in I'}\sum_{j\in J'}(m_{I'j} - m_{I'J'})^2 ,\end{aligned}$$

E_{MSR} represents the variation associated with the interaction between the rows and the columns in the bicluster.

A low (resp. high) E_{MSR} value, i.e., *close* to 0 (resp. higher than a fixed threshold), indicates that the bicluster is strongly (resp. weakly) coherent. If a bicluster has a value of E_{MSR} lower than a given threshold δ then it is called $\delta - bicluster$. However, the E_{MSR} function is inadequate to assess certain types of biclusters [1, 37, 44].

Angiulli *et al.* [3] and Divina and Aguilar-Ruiz [20] propose to use E_{MSR}, the *average row variance*, E_{ARV}, and the *volume* (or *size*), E_V, of a bicluster. The *average row variance* (E_{ARV}) is defined as follows:

$$E_{ARV}(I',J') = \frac{\sum_{i\in I'}\sum_{j\in J'}(m_{ij} - m_{iJ'})^2}{|I'||J'|} \tag{1.12}$$

Let's note that

$$E_{ARV}(I',J') = \frac{\sum_{i\in I'}, \text{Variance of the row } i}{|I'|} \tag{1.13}$$

Biclusters that contain individuals (rows) with large changes in their values for different attributes (columns) are characterized by high values of variance per individual (row). It follows that average row variance can be used to guarantee that a bicluster captures individuals exhibiting coherent trends under some subset of attributes. Let's note that both E_{MSR} and E_{ARV} are E_{AVSS}: Equation 1.10 coincides with Equation 1.11 when $s_{ij} = (m_{ij} - m_{iJ'} - m_{I'j} + m_{I'J'})^2$, and Equation 1.10 coincides with Equation 1.12 when $s_{ij} = (m_{ij} - m_{iJ'})^2$.

The volume $E_V(I', J')$ of a bicluster (I', J') is used to maximize the size of this bicluster. It is defined by:

$$E_V(I', J') = |I'||J'| \tag{1.14}$$

Das *et al.* [16] propose to find the maximum-sized bicluster that does not exceed a certain *coherence* value. The *coherence* is expressed as a MSR score. Hence, Das *et al.* try to maximize the volume $E_V(I', J')$ (Equation 1.14) and find biclusters with a value of E_{MSR} lower than a given threshold δ, for some $\delta \geq 0$ (Equation 1.11).

Teng and Chan [44] propose the *Average Correlation Value* (ACV) function, E_{ACV}, to evaluate the homogeneity of a bicluster. It is defined by the following equation:

$$E_{ACV}(I', J') = \max \left\{ \frac{\sum\limits_{i \in I'} \sum\limits_{j \in I'} |r_{ij}| - |I'|}{|I'|(|I'|-1)}, \frac{\sum\limits_{k \in J'} \sum\limits_{l \in J'} |r_{kl}| - |J'|}{|J'|(|J'|-1)} \right\} \tag{1.15}$$

where:

r_{ij} $(i \neq j)$ is the Pearson's correlation coefficient associated with the row indices i and j in the bicluster (I', J') [35],

r_{kl} $(k \neq l)$ is the Pearson's correlation coefficient associated with the column indices k and l in the bicluster (I', J').

Let us note that the values of E_{ACV} belong to $[0, 1]$. A high (resp. low) E_{ACV} value, i.e., *close* to 1 (resp. *close* to 0), indicates that the bicluster is strongly (resp. weakly) coherent. However, the performance of the E_{ACV} function decreases when noise exists in the data matrix [13]. Some examples are illustrated to assess the E_{ACV} evaluation function [44].

Cheng *et al.* [13] propose to use the E_{ACV} and E_{MSR} functions. A bicluster with a high coherence has a low E_{MSR} value and a high E_{ACV} value.

Ayadi *et al.* propose three evaluation functions:

(i) The first one is the *Average Spearman's Rho* function [4], E_{ASR}. It is defined by the following equation:

$$E_{ASR}(I', J') = 2 \max \left\{ \frac{\sum\limits_{i \in I'} \sum\limits_{j \in I', j \geq i+1} \rho_{ij}}{|I'|(|I'|-1)}, \frac{\sum\limits_{k \in J'} \sum\limits_{l \in J', l \geq k+1} \rho_{kl}}{|J'|(|J'|-1)} \right\} \tag{1.16}$$

where:

ρ_{ij} $(i \neq j)$ is the Spearman's rank correlation associated with the row indices i and j in the bicluster (I', J') [27],

ρ_{kl} $(k \neq l)$ is the Spearman's rank correlation associated with the column indices k and l in the bicluster (I', J').

Let us note that the values of Spearman's rank correlation belong to $[-1..1]$. A high (resp. low) Spearman's rank correlation value, i.e., *close* to 1 (resp. *close* to -1), indicates that the two vectors are strongly (resp. weakly) coherent

[27]. So, the values of the E_{ASR} function belong also to [-1..1]. Hence, a high (resp. low) E_{ASR} value, i.e., *close* to 1 (resp. *close* to -1), indicates that the bicluster is strongly (resp. weakly) coherent. Furthermore, it has been shown that Spearman's rank correlation is less sensitive to the presence of noise in data. Since the E_{ASR} function is based on Spearman's rank correlation, E_{ASR} is also less sensitive to the presence of noise in data.

(*ii*) The second one is the *Average Correspondence Similarity Index* [8], E_{ACSI}. In order to calculate E_{ACSI}, we first discretize the initial data matrix $M(I,J) = [m_{i,j}]$, $I=\{1,2,\ldots,n\}$ and $J=\{1,2,\ldots,m\}$, into a matrix $M' = [m'_{i,l}]$ defined as follows:

$$m'_{i,l} = \begin{cases} 1 & \text{if } m_{i,l} < m_{i,l+1} \\ -1 & \text{if } m_{i,l} > m_{i,l+1} \\ 0 & \text{if } m_{i,l} = m_{i,l+1} \end{cases} \tag{1.17}$$

with $i \in \{1,2,\ldots,n\}$ and $l \in \{1,2,\ldots,m-1\}$.

Let the *Correspondence Similarity List* (CSL) between $gene_i$ and $gene_j$ $(i<j)$, denoted by $CSL_{i,j}$, be the list of all the columns $l \in \{1,2,\ldots,m-1\}$ such that $m'_{i,l} = m'_{j,l}$. Let $NumCSL_{i,j}$ be the number of elements belonging to $CSL_{i,j}$ and $MaxCSL_i = \max\{NumCSL_{i,i+1}, NumCSL_{i,i+2}, \ldots, NumCSL_{i,n}\}$. Let T be a indicator function defined by $T(exp)$=1 when exp is true, and $T(exp)$=0 otherwise. We define the *Correspondence Similarity Index* (E_{CSI}) as follows:

$$E_{CSI}(i,j,k) = \frac{\sum_{l=1}^{m-1} T(m'_{i,l} = m'_{j,l} = m'_{k,l})}{MaxCSL_i} \tag{1.18}$$

with $i \in \{1,2,\ldots,n-2\}$, $j \in \{2,\ldots,n-1\}$, $k \in \{3,\ldots,n\}$ and $i<j<k$.

$E_{CSI}(i,j,k)$ indicates the proportion of trues, i.e., the change in the same direction that exists between rows i, j and k in the same set of columns. This enables to see how genes $gene_i$, $gene_j$ and $gene_k$ behave over a subset of conditions.

Finally, for the whole bicluster, we define the *Average Correspondence Similarity Index* (E_{ACSI}) for the row i ($i \in I'$ and $i<j<k$):

$$E_{ACSI_i}(I',J') = 2\frac{\sum_{j\in I';j\geq i+1} \sum_{k\in I';k\geq j+1} E_{CSI}(i,j,k)}{(|I'|-1)(|I'|-2)} \tag{1.19}$$

(*iii*) The last one is the E_S evaluation function [7]. In order to calculate E_S, we first discretize the initial data matrix $M(I,J)=[m_{i,j}]$, $I=\{1,2,\ldots,n\}$ and $J=\{1,2,\ldots,m\}$, into a matrix $M' = [m'_{i,l}]$ defined as follows:

$$m'_{i,l} = \begin{cases} 1 & \text{if } m_{i,k} < m_{i,q} \\ -1 & \text{if } m_{i,k} > m_{i,q} \\ 0 & \text{if } m_{i,k} = m_{i,q} \end{cases} \tag{1.20}$$

with $i \in \{1,2,..,n\}$, $l \in \{1,2,..,m(m-1)/2\}$, $k \in \{1,2,..,m-1\}$, $q \in \{2,3,..,m\}$ and $q \geq k+1$.

The matrix M' is constructed progressively by merging pairs of columns (conditions) from the input data matrix M. Since M has n rows and m columns, there are $m(m-1)/2$ distinct combinations.

Given a bicluster $b = (I', J')$, the quality of b is assessed *via* the following evaluation function $E_S(b)$:

$$E_S(b) = \frac{\displaystyle\sum_{i \in I'} \sum_{j \in I', j \geq i+1} F_{ij}}{|I'|(|I'|-1)/2} \tag{1.21}$$

with F_{ij} being defined by:

$$F_{ij} = \frac{\displaystyle\sum_{l \in J'} T(m'_{i,l} = m'_{j,l})}{|J'|} \tag{1.22}$$

where $i \in I'$, $j \in I'$ and $j \geq i+1$.

Let us note that $0 \leq F_{ij} \leq 1$. A high (resp. low) F_{ij} value, *close* to 1 (resp. *close* to 0), indicates that the genes $gene_i$ and $gene_j$ (under the given conditions) are strongly (resp. weakly) correlated [7]. Likewise for E_S, a high (resp. low) $E_S(b)$ value, *close* to 1 (resp. *close* to 0), indicates that the bicluster b is strongly (resp. weakly) correlated.

Regarding inter-biclusters evaluation functions, [38] introduced the so-called *Gene Match Score* (GMS) given by:

$$E_{GMS}(B_1, B_2) = \frac{1}{|B_1|} \sum_{(I_1,J_1) \in B_1} \max_{(I_2,J_2) \in B_2} \frac{|I_1 \cap I_2|}{|I_1 \cup I_2|}, \tag{1.23}$$

where B_1, B_2 are two groups of biclusters. Thereafter, more similar measures of match scores have been introduced [2, 13, 22, 29, 36]. For instance, the evaluation functions, herein called *Row and Column Match Scores*, E_{RCMS_1} and E_{RCMS_2}, are proposed in [29] and [22], respectively. E_{RCMS_1} is given by:

$$E_{RCMS_1}(B_1, B_2) = \frac{1}{|B_1|} \sum_{(I_1,J_1) \in B_1} \max_{(I_2,J_2) \in B_2} \frac{|I_1 \cap I_2| + |J_1 \cap J_2|}{|I_1 \cup I_2| + |J_1 \cup J_2|}, \tag{1.24}$$

and E_{RCMS_2} is given by:

$$E_{RCMS_2}(B_1, B_2) = \frac{1}{|B_1|} \sum_{(I_1,J_1) \in B_1} \max_{(I_2,J_2) \in B_2} \frac{|I_1 \cap I_2| + |J_1 \cap J_2|}{|I_1| + |J_1|} \tag{1.25}$$

All these measures of match score are used to assess the accuracy of an algorithm to recover known biclusters and reveal true ones. Both E_{RCMS_1} and E_{RCMS_2} have the advantage of reflecting, simultaneously, the match of the row and column dimensions between biclusters as opposed to E_{GMS} that doesn't take into account column match. They vary between 0 and 1 (the higher the better the accuracy). Let B_{opt} denote the set of true implanted biclusters in the data matrix M and B the set of the output biclusters of a biclustering algorithm. Thus, $E_{GMS}(B_{opt}, B)$ and $E_{RCMS_1}(B_{opt}, B)$ express how well each of the true biclusters are detected by the algorithm under consideration.

$E_{RCMS_2}(B_X, B_Y)$, where B_X (resp. B_Y) denotes the set of biclusters detected by the algorithm X (resp. algorithm Y), has the particularity to allow the quantification of how well each bicluster identified by the algorithm X is contained into some bicluster detected by the algorithm Y.

All the evaluation functions above are deterministic in the sense that no statistical significance is provided for the degree of coherence of any bicluster detected by a biclustering algorithm. When a biclustering algorithm is applied on microarray datasets, enrichment analysis with respect to *Gene Ontology* (GO) annotations, or other specific biological networks like metabolic and protein-protein interaction networks, have usually been applied to check the biological significance of each found bicluster [29, 38, 42]. On a general data matrix M (not necessarily microarray data), Freitas *et al.* [22] recently presented a useful proposal to assign the statistical significance of biclusters. It is given in terms of how "unusually dense" a bicluster is, comparatively, with M. Concretely, the initial data matrix M is transformed into a binary matrix $M_b = [b_{ij}]$ through a discretization defined by

$$b_{ij} = \begin{cases} 1 & , \ m_{ij} \in \mathcal{C} \\ 0 & , \ \text{otherwise} \end{cases} \tag{1.26}$$

where $\mathcal{C}$ is a set of real numbers satisfying the criterion that depends on the strategy defined by the biclustering algorithm. An example of $\mathcal{C}$ applied for the *Iterative Signature Algorithm* (ISA) can be found in [22]. Assuming that the binary matrix M_b contains k 1's, a bicluster B will be considered as *potentially significant*, if its observed number of 1's is significantly greater than the real proportion $p = k/(|I| * |J|)$ of 1's in M. For this statistical testing, the $p-$value is calculated in the following way:

$$p\text{-value}_B = 1 - \phi\Big(\frac{|1_B|/|B| - p}{\sqrt{\frac{p(1-p)}{|B|}}}\Big), \tag{1.27}$$

where ϕ is the standard normal distribution function and $|1_B|$ represents the number of 1's in the bicluster B. Thus, when $p\text{-value}_B < \alpha$, B will be classified as a bicluster *potentially significant* at a level of significance α.
Some biclustering methods identify no more than one bicluster for each application. With this evaluation strategy one can execute several times the biclustering algorithm upon M, and assign for each time a statistical significance for the identified bicluster.

1.5 SYSTEMATIC AND STOCHASTIC BICLUSTERING ALGORITHMS

As we mentioned in the introduction of this chapter, the biclustering problem is NP-hard [14, 30]. Consequently, heuristic search algorithms are typically used to approximate the problem by finding sub-optimal solutions.

We distinguish two main classes of biclustering algorithms: *systematic search* algorithms and *stochastic search* algorithms, also called *metaheuristic* algorithms.

1.5.1 SYSTEMATIC BICLUSTERING ALGORITHMS

Systematic search algorithms are based on one of the following general approaches.

1. *Divide-And-Conquer* (DAC) approach: Generally, this approach divides repeatedly the problem into smaller subproblems with similar structures to the original problem, until these subproblems become smaller enough to be solved directly. The solutions to the subproblems are then combined to create a solution to the original problem. With this approach, we start by a bicluster representing the whole data matrix then we partition this matrix in two submatrices to obtain two biclusters. We reiterate recursively this process until we obtain a certain number of biclusters verifying a specific set of properties. For instance, Prelic *et al.* [38] partition the data matrix M' (M' is a discretization of a data matrix M which contains only binary values where a cell m_{ij} contains 1 if $gene_i$ is expressed under $condition_j$ and 0 otherwise) into three submatrices, one of which contains only 0-cells. The algorithm is then recursively applied to the remaining two submatrices, and ends if the current matrix represents a bicluster which contains only 1's. The advantage of this approach is that it is fast, however, its biggest disadvantage is that it may ignore good biclusters by partitioning them before identifying them. Representative examples of algorithms adopting this approach are given by Dufiy and Quiroz [21], Hartigan [25] and Prelic *et al.* [38].

2. *Greedy Iterative Search* (GIS) approach: This approach constructs a solution in a step-by-step way using a given quality criterion. Decisions made at each step are based on information at hand without worrying about the effect these decisions may have in the future. Moreover, once a decision is taken, it becomes irreversible and is never reconsidered. By applying this approach to the biclustering problem, at each iteration, we construct submatrices of the data matrix by adding/removing a row/column to/from the current submatrix that maximizes/minimizes a certain function. We reiterate this process until no other row/column can be added/removed to/from any submatrix. For instance, the algorithm *Maximum Similarity Biclusters* [29] starts by constructing a

similarity matrix based on a reference gene. A greedy strategy of removing rows/columns iteratively is employed to provide the maximum similarity bicluster in polynomial time. Shabalin *et al.* [40] extract large average submatrices according to a *Bonferroni-based* significance score. Several graph-theoretical approaches have also been proposed. Another recent work was reported by Ayadi *et al.* [8] which constructs a *Directed Acyclic Graph* (DAG) to combine a subset of genes under a subset of conditions iteratively, by adopting the evaluation functions E_{ACSI} and E_{ASR}.

This approach presents the similar advantage and disadvantage of the DAC one. Representative examples of algorithms based on this approach are given by Ben-Dor *et al.* [10], Cheng *et al.* [13], Ihmels *et al.* [26], Liu and Wang [29], Shabalin *et al.* [40], Teng and Chan [44], Yang *et al.* [46, 48] and Ayadi *et al.* [8].

3. *Biclusters Enumeration* (BE) approach: This approach tries to enumerate (explicitly or implicitly) all the solutions for an original problem. The enumeration process is generally represented by a search tree.

   By applying this approach to the biclustering problem, we identify all the possible groups of biclusters in order to keep the *best* one. Ayadi *et al.* [4] use a *Bicluster Enumeration Tree* (BET) to find all the biclusters (nodes) of interest, reachable from the root of the BET, by adopting to the E_{ASR} evaluation function. To reduce the size of the BET, a quality threshold is employed to cut branches that cannot lead to biclusters of desired quality.

   This approach has the advantage of being able to obtain the best solutions. Its disadvantage is that it is costly in computing time and in memory space. Representative examples of algorithms adopting this approach are given by Liu and Wang [28], Tanay *et al.* [42] and Ayadi *et al.* [4, 5].

1.5.2 STOCHASTIC BICLUSTERING ALGORITHMS

Stochastic search algorithm are based on one of the following general approaches.

1. *Neighborhood Search*(NS) approach: Neighborhood search, also called *local search*, is based on the notion of neighborhood and a strategy exploiting this neighborhood. A neighborhood search algorithm starts with an initial solution s and then moves iteratively to a neighboring solution thanks to the neighborhood exploitation strategy. A neighboring solution is generally generated by applying a transformation operator, also called *move operator*, to the current solution. At each iteration, the neighborhood exploitation strategy decides the neighboring solution

to be selected to become the new current solution. For instance, the basic *hill-climbing* strategy replaces the current solution by a neighboring solution of better quality while other strategies based on *simulated annealing* and *tabu search* may substitute the current solution with a worse neighboring solution.

By applying this approach to the biclustering problem, we start by an initial solution which can be a cluster, a bicluster or the whole matrix. Then, at each iteration, we try to improve this solution by adding and/or removing some genes/conditions to minimize/maximize a certain function. The difference with the greedy search algorithms is that if we delete, for example, one gene/condition, we can later add this gene/condition to the solution.

Cheng and Church [14] are probably the first to apply this concept to the biclustering problem. Their goal is to find biclusters with a E_{MSR} value lower than a fixed threshold. Hence, they proposed a local search procedure which deletes/adds genes/conditions to the biclusters. The multiple node deletion method removes all genes and conditions with a E_{MSR} score lower than a fixed threshold. The single node deletion method iteratively removes the gene or column that has low quality according to E_{MSR}. Finally, the node addition method adds genes and conditions that do not decrease the quality of the actual bicluster. In order to find a given number of biclusters, this approach is iteratively executed on the remaining genes and conditions that are not present in the previous obtained biclusters.

The move operator used by Ayadi *et al.* [7] is based on the drop/add operation which removes a $gene_i$ where $i \in I'$ from the bicluster b=(I', J') and adds one $gene_v$, where $v \notin I'$, or various $gene_v$,...,$gene_w$, where $v \notin I', \ldots, w \notin I'$, to b. The move operator can be defined as follows: we first choose a pair of genes $\{gene_i, gene_j\}$ from b which have a bad quality according to an evaluation function. Such a pair of genes contribute negatively to the quality of the bicluster b. Then we look for other pairs of genes $\{gene_j, gene_{r_1}\}$, $\{gene_j, gene_{r_2}\}$, ..., $\{gene_j, gene_{r_n}\}$, where $r_1 \notin I', \ldots, r_n \notin I'$ which have a good quality according to the evaluation function. Hence, $gene_j$ contributes positively to the quality of the bicluster when it is associated with $gene_{r_1} \ldots gene_{r_n}$. Finally, we replace $gene_i$ in b by $gene_{r1}$, $gene_{r2}$,..., $gene_{rn}$. In Das and Idicula [17], the authors generate initial solutions using a K-means clustering algorithm. These solutions are then extended by adding more rows and columns using a *cardinality based greedy randomized adaptive search* procedure. Their greedy strategy makes a choice that optimizes a local gain in the hope that this choice will lead to a globally good solution.

The advantage of this approach lies in the ability to explore large search spaces. This approach also offers the possibility of trade-off between solution quality and running time. Indeed, when the quality of a solution tends to improve gradually over time, the user can stop the execution at a chosen time. The disadvantage of this approach is that the search lead to sub-optimal solutions (local maxima). Representative examples of algorithms adopting this approach are given by Bryan *et al.* [12], Cheng and Church [14], Dharan and Nair [18], Das and Idicula [17] and Ayadi *et al.* [7, 6].

2. *Evolutionary Computation* (EC) approach:

   The evolutionary computation approach is based on the natural evolutionary process such as population, reproduction, mutation, recombination, and selection. Candidate solutions of the given problem are sampled by a set of individuals in a population. An evaluation mechanism (fitness evaluation) is established to assess the quality of each individual. Evolution operators eliminate some (less fit) individuals and produce new individuals from selected individuals.

   By applying this approach to the biclustering problem, we start from an initial population of solutions, i.e., clusters, biclusters or the whole matrix, then, we measure the quality of each solution of the population by the fitness function. We select a number of solutions to produce new solutions by recombination and mutation operators. This process ends when a prefixed stop condition is verified. For instance, Divina and Aguilar-Ruiz [19] generate a population representing biclusters of dimension one because these biclusters have a high E_{MSR} score. From this population, selection, crossover and mutation are repeatedly applied to the population. A number of biclusters are selected for reproduction with a tournament selection operator. In other words, a certain number of biclusters are first selected randomly, and the best one according to E_{MSR} is chosen. Each selected pair of parents is recombined by a crossover operator. For this, three crossover (one point, two points and uniform) operators are applied, with equal probability. The resulting offspring is mutated by using three mutation operators: the standard mutation operator, a mutation operator that adds a row and a column to the bicluster. Since mutation is a highly random operation, it is applied with a low probability. The process is repeated with the new generation of offspring, until a maximum number of generations is reached.

   This approach shares the similar advantages and disadvantages with the neighborhood search approach. Few evolutionary algorithms are reported in the literature for the biclustering problem. Two example are given by Divina and Aguilar-Ruiz [19, 20].

3. *Hybrid* (H) approach: The hybrid approach, also called *memetic approach*, tries to combine both the neighborhood search and the evolutionary approaches. This hybrid approach is known to be quite successful in solving many hard combinatorial search problems. The purpose of such an approach is to take advantage of the complementary nature

of the evolutionary and neighborhood search methods. Indeed, it is generally believed that the evolutionary framework offers more facilities for exploration, while neighborhood search has more capability for exploitation. Combining them may offer a better balance between exploitation and exploration which is highly desirable for an effective search.

Mitra and Banka [33] present a *Multi-Objective Evolutionary Algorithm* (MOEA) based on Pareto dominancy. The authors try to find biclusters with maximum size and homogeneity by using a multi-objective genetic algorithm called NSGA-II (Nondominated sorting genetic algorithm) in combination with the local search procedure. Gallo *et al.* [23] present another hybrid algorithm based on MOEA combined with a local search strategy. They extract biclusters with multiple criteria like maximum rows, columns, homogeneity and row variance. A mechanism for re-orienting the search in terms of row variance and size is provided. The mutation operator is performed when the individual needs to be mutated by means of the probability assigned to the operator. Hence, the gene/condition of the bicluster is mutated at a random position. The crossover operator is applied over both the genes and the conditions. Hence, when both children are obtained by combining at the end and at the center each of the two parents, the individual to select as the only descendant is the non-dominated one. If both are non-dominated, one of them is chosen at random. The authors apply the local search procedure, based on Cheng and Church [14] one, on all the individuals in the resulting population of each generation.

Representative examples of algorithms adopting this approach are given by Bleuler *et al.* [11], Gallo *et al.* [23] and Mitra and Banka [33].

Table 1.2 shows microarray datasets used to evaluate biclustering algorithms. Let us note that some datasets contain missing values. To deal with this problem, Cheng and Church [14] propose to replace the missing values by random ones. Unfortunately, these random values can affect the discovery of coherent biclusters [47]. Another method to deal with missing values, consists in removing the genes/conditions that contain such values [32]. Unfortunately, the removed genes/conditions can affect the discovery of coherent biclusters.

Now, we briefly present some biclustering tools that are publicly available for microarray data analysis.

1. GEMS [45] is a web server for biclustering of microarray data. It is based on a *Gibbs* sampling paradigm [41]. GEMS is available at: http://genomics10.bu.edu/terrence/gems/

2. BicAT [9] is a biclustering analysis toolbox that is mostly used by the community and contains several implementations of biclustering algorithms like the *Order Preserving SubMatrix* (OPSM) algorithm [10], Cheng and Church's algorithm [14], the *Iterative Signature Algorithm*

Table 1.2 Microarray datasets used to evaluate biclustering algorithms.

Dataset	Nbr. genes	Nbr. conditions	Web site
Arabidopsis Thaliana	734	69	http://www.tik.ethz.ch/sop/bimax/
Colon Rectal Cancer	2000	62	http://microarray.princeton.edu/oncology/affydata/index.html
Human B-cell Lymphoma	4026	96	http://arep.med.harvard.edu/biclustering/
Leukemia	7129	72	http://sdmc.lit.org.sg/GEDatasets/Datasets.html
Lung Cancer	12,533	181	http://sdmc.lit.org.sg/GEDatasets/Datasets.html
Ovarian Cancer Tumour	15,154	253	http://sdmc.lit.org.sg/GEDatasets/Datasets.html
Prostate Cancer	12,600	136	http://sdmc.lit.org.sg/GEDatasets/Datasets.html
Saccharomyces Cerevisiae	2993	173	http://www.tik.ethz.ch/sop/bimax/
Yeast Cell Cycle	2884	17	http://arep.med.harvard.edu/biclustering/

(ISA) [26], the *xMotif* algorithm [34] and the *Bimax* algorithm [38]. BicAT is available at:
http://www.tik.ee.ethz.ch/sop/bicat

3. *BiVisu* [13] is a biclustering algorithm based on *Parallel Coordinate* (PC) formulation. It can visualize the detected biclusters in a 2D setting by using PC plots. *BiVisu* is available at :
http://www.eie.polyu.edu.hk/nflaw/Biclustering/index.html

4. *Bayesian BiClustering* model (BBC) [24] is a biclustering algorithm based on Monte Carlo procedure. BBC is available at:
http://www.people.fas.harvard.edu/junliu/BBC/

5. *BicOverlapper* [39] is a visual framework that supports:
- Simultaneous visualization of one or more sets of biclusters,
- Visualization of microarray data matrices as heatmaps and PC,
- Visualization of *transcription regulatory networks*,
- And, linkage of different visualizations and data to achieve a broader analysis of an experiment results.
BicOverlapper is available at:
http://vis.usal.es/bicoverlapper/

6. *e-CCC-Biclustering* [31] is a biclustering algorithm that can find and report all maximal contiguous column coherent biclusters with approximate expression patterns. *e-CCC-Biclustering* is available at:
http://kdbio.inesc-id.pt/software/e-ccc-biclustering

1.6 BICLUSTERS VALIDATION

Biological validation can qualitatively evaluate the capacity of an algorithm to extract meaningful biclusters from a biological point of view. Assessing the biological meaning of the results of a biclustering algorithm is not a trivial task because there do not exist general guidelines in the literature on how to achieve this task. Several authors use artificial datasets to validate their approaches. However, artificial scenario is inevitably biased regarding the underlying model and only reflects certain aspects of biological reality.

One possible validation concerns the *coverage* of biclusters. In fact, in the biclustering domain, it is interesting to have a good compromise between the size and the coherence of a bicluster. However, this is not enough to have a good group of biclusters. It is very important that the final group of biclusters provides good coverage of the dataset. A large coverage of the dataset is very important in several applications that employ biclusters. Indeed, the higher the number of highlighted correlations is, the greater the amount of extracted information is.

To assess the biclusters biologically, we can use *Gene Ontology* (GO) annotation [15]. In GO, genes are assigned to three structured, controlled vocabularies, i.e., *ontologies*, that describe genes products in terms of associated *biological processes, cellular components* and *molecular functions* in a species-independent manner. Users measure the degree of enrichment, i.e., p-value, by using a cumulative *hypergeometric* distribution that involves the probability of observing the number of genes from a particular GO category, i.e., *biological processes*, *cellular components* and *molecular functions*, within each bicluster. Statistical significance is evaluated for the genes in each bicluster by computing p-values which indicate how well they match with the different GO categories. Let us note that a smaller p-value, *close* to 0, is indicative of a better match [43].

The *Gene Ontology Consortium* (GOC) [15] (http://www.geneontology.org) is involved in the development and application of the GO. In the following, we present examples of web-tools related to GOC.

1. *GO Term Finder* (http://db.yeastgenome.org/cgi-bin/GO/goTermFinder) searches for significant shared GO terms, or parents of GO terms, used to annotate genes products in a given list.

2. *FuncAssociate* (http://llama.med.harvard.edu/cgi/func/funcassociate) is a web-based tool that accepts as input a list of genes and returns a list of GO attributes that are over/under-represented among the genes of the input list. Only those over/under-represented genes are reported.

3. *GENECODIS* (http://genecodis.dacya.ucm.es/) is a web-based tool for the functional analysis of a list of genes. It integrates different sources of information to search for annotations that frequently co-occur in a list of genes and rank them according to their statistical significance.

4. *GeneBrowser* (http://bioinformatics.ua.pt/genebrowser2/) is a web-based tool that, for a given list of genes, combines data from several public databases with visualisation and analysis methods to help identify the most relevant and common biological characteristics. The functionalities provided include the following: a central point with the most relevant biological information for each inserted gene; a list of the most related papers in PubMed[1] and gene expression studies in *ArrayExpress*; and an extended approach to functional analysis applied to GO, homologies, gene chromosomal localisation and pathways.

The microarray datasets presented in Table 1.2 are such that each experimental condition corresponds to a patient presenting a kind of pathology. For example, the *Leukemia* dataset discriminates patients affected by either *Lymphoblastic* or *Myeloid leukemia*. Thus, we do not know the biological coherence between genes, while we know the medical classification of conditions. In this case, we can evaluate the ability of an algorithm to separate the samples according to their known classification. To this end, we can compute the

[1]http://www.ncbi.nlm.nih.gov/pubmed

number of columns labelled with the same class and belonging to the same bicluster. Obviously, the higher the number of columns in a bicluster labelled with the same class label is, the higher its biological quality is. In fact, this means that many patients with the same diagnosis are grouped together with respect to a subset of genes, thus we could induce that those genes probably have similar functional category and characterize the majority class of patients.

1.7 CONCLUSION

The biclustering of microarray data has been the subject of a large research, no one of the existing biclustering algorithms is perfect and the construction of biologically significant groups of biclusters for large microarray data is still a problem that requires a continuous work.

Biological validation of biclusters of microarray data is one of the most important open issues. So far, there are no general guidelines in the literature on how to validate biologically such biclusters.

References

1. J. S. Aguilar-Ruiz. Shifting and scaling patterns from gene expression data. *Bioinformatics*, 21:3840–3845, 2005.
2. W. Ahmad and A. Khokhar. chawk: An efficient biclustering algorithm based on bipartite graph crossing minimization. In *Proceedings of the 2007 VLDB Workshop on Data Mining in Bioinformatics*, New York, NY, USA, 2007. ACM.
3. F. Angiulli, E. Cesario, and C. Pizzuti. Random walk biclustering for microarray data. *Journal of Information Sciences*, pages 1479–1497, 2008.
4. W. Ayadi, M. Elloumi, and J. K. Hao. A biclustering algorithm based on a bicluster enumeration tree : Application to DNA microarray data. *BioData Mining*, 2(9), 2009.
5. W. Ayadi, M. Elloumi, and J.K. Hao. Bimine+: An efficient algorithm for discovering relevant biclusters of DNA microarray data. *Submitted*.
6. W. Ayadi, M. Elloumi, and J.K. Hao. Pattern-driven neighborhood search for biclustering of microarray data. *Submitted*.
7. W. Ayadi, M. Elloumi, and J.K. Hao. Iterated local search for biclustering of microarray data. In *The 5th Pattern Recognition in Bioinformatics, PRIB'10. Lecture Notes in BioInformatics (LNBI)*, pages 219–229. Springer-Verlag, 2010.
8. W. Ayadi, M. Elloumi, and J.K. Hao. Bicfinder: a biclustering algorithm for microarray data analysis. *To appear in Knowledge and Information Systems: An International Journal*, 2011.

9. S. Barkow, S. Bleuler, A. Prelic, P. Zimmermann, and E. Zitzler. Bicat: a biclustering analysis toolbox. *Bioinformatics*, 22(10):1282–1283, 2006.

10. A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini. Discovering local structure in gene expression data: the order-preserving submatrix problem. In *RECOMB '02: Proceedings of the sixth annual international conference on Computational biology*, pages 49–57, New York, NY, USA, 2002. ACM.

11. S. Bleuler, A. Prelic, and E. Zitzler. An ea framework for biclustering of gene expression data. In *Proceedings of Congress on Evolutionary Computation*, pages 166–173, 2004.

12. K. Bryan, P. Cunningham, and N. Bolshakova. Application of simulated annealing to the biclustering of gene expression data. In *IEEE Transactions on Information Technology on Biomedicine, 10(3)*, pages 519–525, 2006.

13. K. O. Cheng, N. F. Law, W. C. Siu, and A. W. Liew. Identification of coherent patterns in gene expression data using an efficient biclustering algorithm and parallel coordinate visualization. *BMC Bioinformatics*, 9(210):1282–1283, 2008.

14. Y. Cheng and G. M. Church. Biclustering of expression data. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93–103. AAAI Press, 2000.

15. Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nat. Genet*, 25:25–29, 2000.

16. R. Das, S. Mitra, H. Banka, and S. Mukhopadhyay. Evolutionary biclustering with correlation for gene interaction networks. In *PReMI, LNCS*, pages 416–424, 2007.

17. S. Das and S. M. Idicula. Application of cardinality based grasp to the bi clustering of gene expression data. *International Journal of Computer Applications*, 1(18), 2010.

18. A. Dharan and A. S. Nair. Biclustering of gene expression data using reactive greedy randomized adaptive search procedure. *BMC Bioinformatics*, 10(Suppl 1):S27, 2009.

19. F. Divina and J. S. Aguilar-Ruiz. Biclustering of expression data with evolutionary computation. *IEEE Transactions on Knowledge & Data Engineering*, 18(5):590–602, 2006.

20. F. Divina and J. S. Aguilar-Ruiz. A multi-objective approach to discover biclusters in microarray data. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 385–392, New York, NY, USA, 2007. ACM.

21. D. Dufiy and A. Quiroz. A permutation based algorithm for block clustering. *Journal of Classification*, (8):65–91, 1991.

22. A. Freitas, V. Afreixo, M. Pinheiro, J. L. Oliveira, G. Moura, and M. Santos. Improving the performance of the iterative signature algorithm for the identification of relevant patterns. *Statistical Analysis and Data Mining*, 4(1):71–83, 2011.

23. C. A. Gallo, J. A. Carballido, and I. Ponzoni. Microarray biclustering: A novel memetic approach based on the pisa platform. In *EvoBIO '09: Proceedings of the*

7th European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, pages 44–55, Berlin, Heidelberg, 2009. Springer-Verlag.

24. J. Gu and J. S. Liu. Bayesian biclustering of gene expression data. *BMC Genomics*, 9(Suppl 1):S4, 2008.
25. J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, 67(337):123–129, 1972.
26. J. Ihmels, S. Bergmann, and N. Barkai. Defining transcription modules using large-scale gene expression data. *Bioinformatics*, 13:1993– 2003, 2004.
27. E. L. Lehmann and H. J. M. D'Abrera. Nonparametrics: Statistical methods based on ranks. *rev. ed. Englewood Cliffs, NJ: Prentice-Hall*, pages 292–323, 1998.
28. J. Liu and W. Wang. Op-cluster: Clustering by tendency in high dimensional space. *IEEE International Conference on Data Mining*, pages 187–194, 2003.
29. X. Liu and L. Wang. Computing the maximum similarity bi-clusters of gene expression data. *Bioinformatics,*, 23(1):50–56, 2007.
30. S C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 1(1):24–45, 2004.
31. S.C. Madeira and A. L. Oliveira. A polynomial time biclustering algorithm for finding approximate expression patterns in gene expression time series. *Algorithms for Molecular Biology*, 4:8, 2009.
32. S.C. Madeira and A.L. Oliveira. An efficient biclustering algorithm for finding genes with similar patterns in time-series expression data. In *Proc. of the 5th Asia Pacific Bioinformatics Conference, Series in Advances in Bioinformatics and Computational Biology. Volume 5.Imperial College Press*, pages 67–80, 2007.
33. S. Mitra and H. Banka. Multi-objective evolutionary biclustering of gene expression data. *Pattern Recogn.*, 39(12):2464–2477, 2006.
34. T. M. Murali and S. Kasif. Extracting conserved gene expression motifs from gene expression data. *Pac. Symp. Biocomput*, 8:77–88, 2003.
35. J. L. Myers and D. W. Arnold. Research design and statistical analysis. 2003.
36. Y. Okada, K. Okubo, P. Horton, and W. Fujibuchi. Exhaustive search method of gene expression modules and its application to human tissue data. In *IAENG International Journal of Computer Science, 34*, pages 1–16, 2007.
37. B. Pontes, F. Divina, R. Giráldez, and J. S. Aguilar-Ruiz. Virtual error: A new measure for evolutionary biclustering. In E. Marchiori, J. H. Moore, and J. C. Rajapakse, editors, *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, volume 4447 of *Lecture Notes in Computer Science*, pages 217–226. Springer, 2007.
38. A. Prelic, S. Bleuler, P. Zimmermann, P. Buhlmann, W. Gruissem, L. Hennig, L. Thiele, and E. Zitzler. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, 22(9):1122–1129, 2006.

39. R. Santamaria, R. Thern, and L. Quintales. A visual analytics approach for understanding biclustering results from microarray data. *BMC Bioinformatics*, 9:247, 2008.

40. A.A. Shabalin, V. J. Weigman, C. M. Perou, and A. B. Nobel. Finding large average submatrices in high dimensional data. *Annals of Applied Statistics*, 3(985), 2009.

41. Q. Sheng, Y. Moreau, and B. De Moor. Biclustering microarray data by gibbs sampling. *Bioinformatics*, 19:II196–II205, 2003.

42. A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18:S136–S144, 2002.

43. S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church. Systematic determination of genetic network architecture. *Nature Genetics*, 22:281–285, 1999.

44. L. Teng and L. Chan. Discovering biclusters by iteratively sorting with weighted correlation coefficient in gene expression data. *J. Signal Process. Syst.*, 50(3):267–280, 2008.

45. C. J. Wu and S. Kasif. Gems: a web server for biclustering analysis of expression data. *Nucleic Acids Research*, 33:W596–W599, 2005.

46. J. Yang, H. Wang, W. Wang, and P. Yu. Enhanced biclustering on expression data. In *BIBE '03: Proceedings of the 3rd IEEE Symposium on BioInformatics and BioEngineering*, page 321, Washington, DC, USA, 2003. IEEE Computer Society.

47. J. Yang, W. Wang, H. Wang, , and P. S. Yu. deltaclusters: Capturing subspace correlation in a large data set. In *Proceedings of the 18th IEEE Conference on Data Engineering*, page 517528, 2002.

48. Y. H. Yang, S. Dudoit, P. Luu, D. M. Lin, V. Peng, J. Ngai, and T. P. Speed. Normalization for cdna microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Res*, 30:1–12, 2002.

49. Z. Zhang, A. Teo, B. C. Ooi, and K. L. Tan. Mining deterministic biclusters in gene expression data. *Bioinformatic and Bioengineering, IEEE International Symposium on*, 0:283, 2004.