Analysis of Gene Expression Data **Spring Semester, 2004**

Lecture 8: May 13, 2004

Lecturer: Amos Tanay *Scribe: Michal Ozery-Flato and Israel Steinfeld*[1]

8.1 Elements of Bicluser Algorithms

As we have seen in previous lectures, the technology of DNA chips allows the measurement of mRNA levels simultaneously for thousands of genes. The results of DNA chip experiments are usually organized together in a *gene expression matrix*, with rows corresponding to genes and columns corresponding to conditions. A *bicluster* can be defined as a submatrix spanned by a subset of genes and a subset of conditions in which the genes have joint behavior in the corresponding conditions. Different algorithmic approaches to the biclustering problem use different measures for the quality of a given biclustering solution. Hence, the definition of the goal function of each algorithm is part of its description. There are two important themes regarding biclustering algorithms:

1. Data normalization. A common preprocessing step before applying any clustering algorithm is to normalize the rows or columns of the gene expression matrix, depending on the dimension on which the clustering is applied (rows/columns). In biclustering algorithms, for which the two dimensions are considered, the question of how normalization should be done is not simple. Some of the biclustering algorithms do not refer to the normalization problem and assume that the given data is normalized.

2. Eliminating redundancies when searching for all (or many) biclusters. While in clustering solutions each gene/condition participates in at most one cluster, in biclustering solutions genes and conditions can be part of several biclusters. A bicluster algorithm should eliminate redundancies ("similar" biclusters) from its output.

This lecture can be viewed as a direct continuation of Lecture 6 which already described the notion of biclustering and presented two biclustering algorithms. These algorithms (referred here as "previous algorithms") have each a different definition for a bicluster.

This lecture is organized as follows. In Section 8.2 we briefly review the two different ways the previous biclustering algorithms regarded the notion of biclustering. We then describe two additional biclustering algorithms. In Section 8.3 we describe the Iterative Signature Algorithm which is a simple randomized biclustering algorithm. In Section 8.4 we describe SAMBA – a fast algorithm which identifies biclusters of high statistical quality using combinatorial algorithms.

[1]Based in part on a scribe by Irit Gat and Amos Tanay, May 2002.

8.2 Previous biclustering algorithms

For each of the previous algorithms, we present here a brief description of the set of biclusters the algorithm is aimed to find. For a full description of these algorithms, the reader is referred to Lecture 6.

8.2.1 Cheng and Church's algorithm

Cheng and Church [4] implicitly assume that (gene, condition) pairs in a "good" bicluster have a constant expression level, plus possibly additive row and column specific effects. After removing row, column and submatrix averages, the residual level should be as small as possible. More formally, given the gene expression matrix E, a subset of genes I and a subset of conditions J, we define $e_{Ij} = \frac{\sum_{i \in I} e_{ij}}{|I|}$ (column subset average) $e_{iJ} = \frac{\sum_{j \in J} e_{ij}}{|J|}$ (row subset average) and $e_{IJ} = \frac{\sum_{i \in I, j \in J} e_{ij}}{|I||J|}$ (submatrix average). We define the *residue score* of an element e_{ij} in a submatrix E_{IJ} as $RS_{IJ}(i,j) = e_{ij} - e_{Ij} - e_{iJ} + e_{IJ}$ and the *mean square residue score* of the entire submatrix as $H(I,J) = \sum_{i \in I, j \in J} \frac{RS_{IJ}(i,j)^2}{|I||J|}$. Given the score definition, the *maximum bicluster problem* seeks a bicluster of maximum size among all biclusters with score not exceeding a threshold δ. The size can be defined in several ways, for example as the number of cells in the matrix ($|I||J|$) or the number of rows plus number of columns ($|I| + |J|$).

To discover more than one bicluster, Cheng and Church suggested repeated application of the biclustering algorithm on modified matrices. The modification includes randomization of the values in the cells of the previously discovered biclusters, preventing the correlative signal in them to be beneficial for any other bicluster in the matrix. This has the obvious effect of precluding the identification of biclusters with significant overlaps.

8.2.2 Coupled two-way clustering

Coupled two-way clustering (CTWC), introduced by Getz, Levine and Domany [8] does not define any goal function or other measure for the quality of a given biclustering solution. Instead it defines a generic scheme for transforming a one-dimensional clustering algorithm into a biclustering algorithm. The algorithm relies on having a one-dimensional (standard) clustering algorithm that can discover significant (termed *stable* in [8]) clusters. In the context of the CTWC algorithm, significant gene clusters and significant condition clusters are referred as *stable gene sets* and *stable conditions sets* respectively. The submatrices defined by pairings of stable gene sets and stable condition sets are called *stable submatrices* and correspond to biclusters. The CTWC algorithm recursively applies the one-dimensional algorithm to stable submatrices to produce (if exist) new stable gene sets and stable condition sets.

8.3 The Iterative Signature Algorithm

In the Iterative Signature Algorithm (ISA) [12, 3] the notion of a significant bicluster is defined intrinsically on the bicluster genes and conditions – the conditions of a bicluster uniquely define the genes and vice versa. The intuition is that the genes in a bicluster are co-regulated and, thus, for each condition the average gene expression over all the biclsuter's genes should be surprising (unusually high or low) and for each gene the average gene expression over all biclusters conditions should be surprising. This intuition is formalized using a simple linear model for gene expression assuming normally distributed expression levels for each gene or condition as shown below.

8.3.1 Normalization

The algorithm, presented in Figure 8.1, uses two normalized copies, E^G and E^C, of the original gene expression matrix E. The matrix E^G (E^C) results from E by normalizing each column (row) so that for every condition (gene) the expression level of a random gene (condition) has mean 0 and variance 1.

8.3.2 Bicluster definition

Suppose $G' = \{g\}$ is a set of independent random variables, where each random variable $g \in G$ has mean 0 and variance 1. In this case, the average of all the random variables in G' (i.e. $\frac{1}{|G'|}\sum_{g\in G'} g$) has a mean and a variance equal to 0 and $\frac{1}{|G'|}$ respectively. For $G' \subseteq G$ and $C' \subseteq C$ we define:

$$e^G_{G'c} = \frac{1}{|G'|}\sum_{g\in G'} E^G_{cg} \tag{8.1}$$

$$e^C_{gC'} = \frac{1}{|C'|}\sum_{c\in C'} E^C_{cg} \tag{8.2}$$

Note that $e^G_{Gc} = e^C_{gC} = 0$ due to the normalization made in E^G and E^C.

For a fixed C' (G') let σ_G (σ_C) be the standard deviation of the means $\{e^C_{gC'} : \ g \in G\}$ ($\{e^G_{G'c} : \ c \in C\}$). The standard deviation σ_G (σ_C) can be predicted as $\frac{1}{\sqrt{|C'|}}$ ($\frac{1}{\sqrt{|G'|}}$) as $e^C_{gC'}$ ($e^G_{G'c}$) being a linear sum of $|C'|$ ($|G'|$) independent standard random variables. Alternatively, the standard deviations, σ_G and σ_C, can be estimated directly from the data, correcting for possible biases in the statistics of the specific condition and gene sets used.

Given a set of genes G' and a threshold parameter T_C define:

$$\mathrm{ISA}(G') = \{c \in C : \ |e^G_{G'c}| > T_C\sigma_C\} \tag{8.3}$$

Given a set of conditions C' and a threshold parameter T_G define:

$$\mathrm{ISA}(C') = \{g \in G : \ |e^C_{gC'}| > T_G \sigma_G\} \tag{8.4}$$

A (perfect) bicluster $B = (G', C')$ is required to have:

$$\mathrm{ISA}(G') = C', \quad \mathrm{ISA}(C') = G' \tag{8.5}$$

The idea is that if the genes in G' are up- or down-regulated in the conditions C' then their average expression should be significantly far (i.e., T_C standard deviations) from its expected value on random matrices (which is 0 since the matrix is standardized). A similar argument holds for the conditions in C'. In other words, in a bicluster, the z-score of each gene, measured w.r.t. the bicluster's conditions, and the z-score of each condition, measured w.r.t. the bicluster's genes, should exceed a threshold. As we shall see below, ISA will not discover biclusters for which the conditions (8.5) hold strictly, but will use a relaxed version.

8.3.3 Searching for biclusters

Define a directed bipartite graph $B(\mathcal{G}, \mathcal{C}, E)$ as follows. $\mathcal{G} = \{G' : \ G' \subseteq G\}$, $\mathcal{C} = \{C' : \ C' \subseteq C\}$, $E = \{(G', C') : \ \mathrm{ISA}(G') = C', G' \subseteq G, C' \subseteq C\} \cup \{(G'', C'') : \ \mathrm{ISA}(C'') = G'', G'' \subseteq G, C'' \subseteq C\}$. Following the conditions (8.5), a (perfect) bicluster is a cycle of size 2. However, as we mentioned before, the algorithm searches for an approximation of a (perfect) bicluster.

The algorithm starts from an arbitrary set of genes $G_0 = G_{in}$. The set may be randomly generated or selected based on some prior knowledge. The algorithm then repeatedly applies the update equations:

$$C_i = \mathrm{ISA}(G_i), \quad G_{i+1} = \mathrm{ISA}(C_i) \tag{8.6}$$

Let g_n be a binary vector of size $|G|$ for which $g_n^{(i)} = 1$ iff $i \in G_n$. An approximated bicluster is a pair (G_n, C_n) satisfying:

$$\frac{|g_n - g_{n-i}|}{|g_n + g_{n-i}|} < \epsilon \tag{8.7}$$

for all $i > 0$ smaller than some m. The ISA thus searches for a series of gene sets $\{G_i\}$ which converges to an approximated fixed point that is considered to be a bicluster. The actual fixed point depends on both the initial set G_{in} and the threshold parameters T_C, T_G.

The ISA algorithm can be generalized by assigning weights for each gene/condition such that genes/conditions with a significant behavior (higher z-score) will have larger weights. In this case, there are two changes to the algorithm:

1. the simple means, $e^G_{G'c}$ and $e^C_{gC'}$, are replaced by weighted means.

2. The vector g_n (see 8.7) has real-valued elements and is defined as follows: $g_n^{(i)} = \mathrm{weight}(i)$ if $i \in G_n$, otherwise $g_n^{(i)} = 0$.

8.3.4 Applications

To generate a representative set of biclusters, it is possible to do one of the following:

1. run ISA with different seeds ($\{G_{in}\}$), including known sets of associated genes or random sets. When a random set is chosen there is a possibility that the algorithm will end reporting an empty bicluster. Starting from genes with a known functional annotation or transcription factor can lead to a bicluster containing the original set of genes with other new genes having similar characteristics. To find similar genes from different species, the algorithm can start from a known set of orthologs.

2. Run ISA with variant thresholds (see 8.3 and 8.4). This action affects the ISA function and consequently the traversed bipartite graph. Scanning over different values for (T_C, T_G) reveals the modular structure at different resolutions: Lower thresholds yield larger biclusters whose coregulation is relatively loose, while higher thresholds lead to smaller, tightly coregulated biclusters.

After eliminating redundancies (similar solutions), the set of fixed points can be analyzed as a set of biclusters.

8.3.5 Discussion on ISA

ISA is an elegant algorithm which have a quite fast implementation. It offers a satisfying simple solution to the normalization problem. It is known to have good empirical results in several cases. However, there are some difficulties in the use of ISA:

1. There is no schema for choosing the values for thresholds.

2. Finding good seeds for the algorithm is not a simple task either.

3. We cannot know if we found all or even most of the exiting (approximated) biclusters.

4. Since the algorithm use Z-score statistics, it implicitly assumes that the data (genes and conditions) have normal distribution. However, if the data have non-normal behavior, then the algorithm may fail. For instance, assume that the data contains a gene g and a condition c for which the expression level is much higher than all the other levels (E_{gc} is extremely high). In this case, if a set of genes G' contains g then $c \in \mathrm{ISA}(G')$ and vice versa: $c \in C'$ implies $g \in \mathrm{ISA}(C')$. Hence, we can get a bicluster in which there are a condition and a gene which have no relevance to the other genes/conditions in the bicluster.

ISA(G, C, E, G_{in}, T_G, T_C, m, ϵ):
* *G : genes. C : conditions. E : Gene expression matrix.*
G_{in} : Initial gene set.
T_G, T_C: gene and condition z-score thresholds.
m, ϵ: stopping criteria. *\
Construct matrix E^C.
Construct matrix E^G.
Initialize counter $n = 0$.
Initialize the current genes set $G' = G_{in}$ and compute g_0 based on G'
While (**true**) do
    Compute $e^G_{G'c}$ for $c \in C$.
    $C' = \{c \in C : \ |e^G_{G'c}| > \frac{T_C}{\sqrt{|G'|}}\}$
    Compute $e^C_{gC'}$ for $g \in G$.
    $G' = \{g \in G : \ |e^C_{gC'}| > \frac{T_G}{\sqrt{C'}}\}$
    $n = n + 1$
    Compute g_n from G'.
    If $(n < m - 1)$ **continue**.
    Initialize counter $i = 1$
    while $i < m$
        If $(\frac{|g_n - g_{n-i}|}{|g_n + g_{n-i}|} \geq \epsilon)$ **break**
        $i = i + 1$
    If $i == m$ **break**
Report G', C'

Figure 8.1: The ISA algorithm for finding a single bicluster.

8.4 SAMBA

We next describe SAMBA, the biclustering algorithm of [15]. SAMBA stands for Statistical Algorithmic Method for Bicluster Analysis. The motivation underlying its development is the need for a fast biclustering method that would produce statistically significant results as part of its design. The key point in the understanding of SAMBA is the statistical model used by the bicluster scoring scheme. The hope is that a model which captures the essential features of the

data would guarantee high quality results.

8.4.1 Data Modeling

The intuitive notion of a bicluster is a subset of genes that exhibit similar expression patterns over a subset of conditions. Following this intuition we define a bicluster as a subset of genes that *jointly respond* across a subset of conditions, where a gene is termed responding in some condition if its expression level changes significantly at that condition w.r.t. its normal level.

The expression data is modeled as a bipartite graph whose two parts correspond to conditions and genes, respectively, with edges for significant expression changes. Later, we shall assign weights to the vertex pairs of the bipartite graph according to a statistical model, so that heavy subgraphs would correspond to significant biclusters. We can tag each edge to incorporate the direction of regulation (up or down) as we shall see later. For now assume edges are not tagged.

Formally, given an input gene expression data, we form a bipartite graph $G = (U, V, E)$ (see Figure 8.2 for an example). In this graph, U is the set of conditions, V is the set of genes, and $(u, v) \in E$ iff v responds in condition u, that is, if the expression level of v changes significantly in u. This discretisation can be further developed using a discretisation function, adding edges according to the expression level of v .A bicluster corresponds to a subgraph $H = (U', V', E')$ of G, and represents a subset V' of genes that are co-regulated under a subset of conditions U' (see Figure 8.2). The *weight* of a subgraph (bicluster) is the sum of the weights of gene-condition pairs in it, including edges and non-edges.

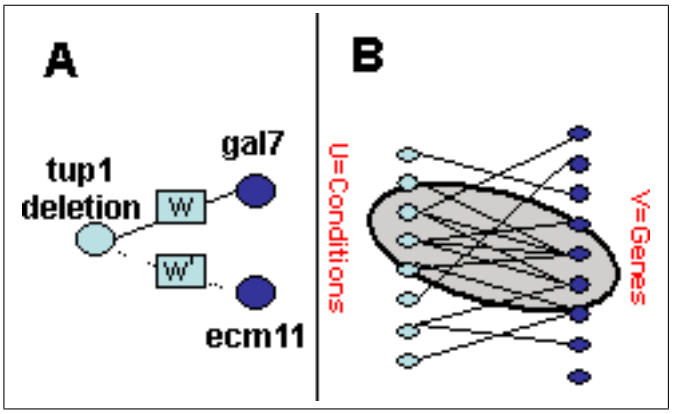


Figure 8.2: Source: [15]. SAMBA model : Gene expression data is modeled using a bipartite graph whose two sides correspond to the set of conditions U and the set of genes V. An edge (u, v) indicates the response of gene v in condition u. A statistical model assigns weights to the edges and non-edges of the graph. A: Part of the graph showing the condition "tup1 deletion" and its effect on the genes "gal7" (response) and "ecm11" (no response). B: A heavy subgraph (shaded) representing a significant bicluster.

In order to assign statistical meaning to the weight of a subgraph, the authors developed

statistical models for the bipartite graph representation of expression data. Using these models one can derive scoring schemes for assessing the significance of an observed subgraph (corresponding to a bicluster). This is done so that the score can be expressed as a sum of independent contributions from each of the node pairs (condition-genes) in the subgraph. Using this model, we can reduce the biclustering problem to the problem of finding heavy subgraphs in a bipartite graph.

8.4.2 A Simple Model

The first statistical model we present is a simplistic one, and is presented as a motivation for the more sophisticated model that will follow. Let $H = (U', V', E')$ be a subgraph of G. Denote $|U'| = m', |V'| = n', |E'| = k'$. The model assumes that edges occur independently with equal probably p, where $p = \frac{|E|}{|U||V|}$ (graph density). Denote by $BT(k, p, n)$ the binomial tail, i.e., the probability of observing k or more successes in n trials, where each success occurs independently with probability p. Then the probability of observing a graph at least as dense as H according to this model is $P(H) = BT(k', p, n'm') = \sum_{k''=k'}^{n'm'} \binom{n'm'}{k''} p^{k''}(1-p)^{n'm'-k''}$.

Our goal is to find a subgraph H with the lowest $P(H)$. By bounding the terms of the binomial tail using the first term (which is the largest, assuming that $p < 1/2$), we obtain the following upper bound for $P(H)$: $P^*(H) \leq p^{k'}(1-p)^{n'm'-k'} \sum_{k''=k'}^{n'm'} \binom{n'm'}{k''} \leq 2^{n'm'} p^{k'}(1-p)^{n'm'-k'}$. Seeking a subgraph H minimizing $\log P^*(H)$ is equivalent to finding a maximum weight subgraph of G where each edge has positive weight $(-1 - \log p)$ and each non-edge has negative weight $(-1-\log(1-p))$ since the weight of a subgraph are $log(P^*(H'))$ under these weights.

8.4.3 A Refined Model

The simple model presented above is far from the reality. In fact, if we look at the degree distribution of real gene expression data we can identify a very non uniform behavior, where some of the conditions and genes have very high degrees and others very low degrees. Assuming a simple random graph model would result in very high scores for the "bicluster" defined by all high degree nodes in the graph. We next describe a refined null model that takes into account the variability of the degrees in G, i.e., it incorporates the characteristic behavior of each specific condition and gene. This behavior may reflect some experiment artifact (some genes are more likely to have noisy measurements) or some biological actual meaning (some of the genes participate in many of the cellular pathways).

Let $H = (U', V', E')$ be a subgraph of G and denote $\overline{E'} = (U' \times V') \setminus E'$. For a vertex $w \in U' \cup V'$ let d_w^G denote its degree in G. The refined null

model assumes that the occurrence of each edge (u, v) is an independent Bernoulli variable with parameter $p_{(u,v)}$. The probability $p_{(u,v)}$ is the fraction of bipartite graphs with degree

sequence identical to G that contain the edge (u, v), or

$$p_{(u,v)} = \frac{|\{G' = (U, V, E') | \forall w, d_w^{G'} = d_w^G, (u, v) \in E'\}|}{|\{G' = (U, V, E') | \forall w, d_w^{G'} = d_w^G\}|} \tag{8.8}$$

We can estimate $p_{(u,v)}$ using a Monte-Carlo like process, starting from the original graph and performing a sequence of random edge swaps that preserve the degrees (A formal proof of a sampling lemma in the space of fixed degree graphs is however still an open problem). The probability of observing H is thus $P(H) = (\prod_{(u,v)\in E'} p_{(u,v)}) \cdot (\prod_{(u,v)\in \overline{E'}} (1 - p_{(u,v)}))$. However, we cannot simply compare subgraphs according to this probability, since it improves (decreases) as the size of H increases.

To overcome this problem, we use a likelihood ratio to capture the significance of biclusters. The null model is as stated above. For the alternative model we assume that each edge of a bicluster occurs with constant probability $p_c > \max_{(u,v)\in U\times V} p_{(u,v)}$. This model reflects our belief that biclusters represent approximately uniform relations between their elements. The log likelihood ratio for H is therefore:

$$\log L(H) = \sum_{(u,v)\in E'} \log \frac{p_c}{p_{(u,v)}} + \sum_{(u,v)\in \overline{E'}} \log \frac{1 - p_c}{1 - p_{(u,v)}}$$

Setting the weight of each edge (u, v) to $\log \frac{p_c}{p_{(u,v)}} > 0$ and the weight of each non-edge (u, v) to $\log \frac{1-p_c}{1-p_{u,v}} < 0$, we conclude that the score of H is simply its weight.

8.4.4 Finding heavy subgraphs

Having assigned weights for edges in our model bipartite graph, such that the weight of a subgraph encode some reasonable likelihood ratio, we now turn to the problem of identifying the maximum likelihood bicluster, which is, under our formulation, the heaviest subgraph.

The computational problem of finding the largest node biclique (the one with the largest number of vertices) in a bipartite graph has an elegant polynomial time algorithm (using matching). Our problem, however, is more closely related to the problem of finding the largest edge biclique (the biclique with the largest number of edges) which is NP-hard for both unweighted [13] and weighted graphs [9]. In fact, the problem of finding the heaviest bipartite subgraph when edges are assigned positive weights and non edges negative weights is hard as well (by reduction from CLIQUE). We thus enforce additional limitation on our graph, namely, restrict the in degree of the genes side, so that a gene that respond in more then d conditions is ignored. This enables us to develop a polynomial algorithm which is later used as the basis for a practical implementation that can avoid the degree restriction. Note that according to the statistical model, genes with high in degree contribute less to the significance of a bicluster, so running the algorithm without them may not be a very restricting limitation.

Maximum Bounded Biclique

We start by describing an $O(|V|2^d)$-time algorithm to find a maximum weight biclique in a bipartite graph whose gene vertices have d-bounded degree. This algorithm will be a key component in the more involved algorithms that follow.

Let $G = (U, V, E)$ be a bipartite graph. We say that G has *d-bounded gene side*, if every $v \in V$ has degree at most d. Let $w : U \times V \to \mathcal{R}$ be a weight function. For a pair of subsets $U' \subseteq U, V' \subseteq V$ we denote by $w(U', V')$ the weight of the subgraph induced on $U' \cup V'$, i.e., $w(U', V') = \sum_{u \in U', v \in V'} w((u, v))$. The *neighborhood* of a vertex v, denoted $N(v)$, is the set of vertices adjacent to v in G. We denote $n = |V|$ throughout.

Problem 8.1 (Maximum Bounded Biclique) Given a weighted bipartite graph G with d-bounded gene side, find a maximum weight complete subgraph of G.

Theorem 8.1 *The maximum bounded biclique problem can be solved in $O(n2^d)$ time and space.*

Proof: Observe that a maximum bounded biclique $H^* = (U^*, V^*, E^*)$ in G must have $|U^*| \leq d$. if not its easy to show that there must be a vertex in U^* that by removing it will give a better biclique, with contradiction to the maximality of H^*. Figure 8.3 describes a hash-table based algorithm that for each vertex $v \in V$ scans all $O(2^d)$ subsets of its neighbors, thereby identifying the heaviest biclique. Each hash entry corresponds to a subset of conditions and records the total weight of edges from adjacent gene vertices. An iteration over subsets of $N(v)$ can be done in $O(2^d)$ time. Hence, the algorithm spends $O(n2^d)$ time on the hashing and finding U_{best}. Since $|U_{best}| \leq d$, computing V_{best} can be done in $O(nd)$ time, so the total running time is $O(n2^d)$. The space complexity is $O(n2^d)$ due to the hash-table. ■

Note that the algorithm can be adapted to give the k condition subsets that induce solutions of highest weight in $O(n2^d \log k)$ time using a priority queue data structure.

Finding Heavy Subgraphs Algorithm

We will now show how to extend the latter algorithm to find heavy subgraphs which are not necessarily complete. For simplicity we shall describe the algorithm assuming that each edge has weight $+1$ and each non-edge has weight -1. Extension to more general weights can be done in a similar manner. Formally, given a bipartite graph $G = (U, V, E)$ define a weight function $w : U \times V \to \{-1, 1\}$ such that $w((u, v)) = 1$ for $(u, v) \in E$, and $w((u, v)) = -1$ for $(u, v) \in (U \times V) \setminus E$. Consider the following problem:

Problem 8.2 (Maximum Bounded Bipartite Subgraph) Given a bipartite graph G with d-bounded gene side, find a maximum weight subgraph of G.

```
MaxBoundBiClique(U, V, E, d):
Initialize a hash table weight; weight_best ← 0
For all v ∈ V do
    For all S ⊆ N(v) do
        weight[S] ←weight[S]+
                   w(S, {v})
        If (weight[S] > weight_best)
          U_best ← S
          weight_best ← weight[S]
Compute V_best = ∩_{u∈U_best} N(u)
Output (U_best, V_best)
```

Figure 8.3: Source: [15]. An algorithm for the maximum bounded biclique problem.

Lemma 8.2 *Let $H^* = (U^*, V^*, E^*)$ be a maximum weight subgraph of G. Then every vertex in H^* is connected to at least half the vertices on the other side of H^*.*

Proof: Follows from the choice of weights, since if a vertex $v \in V^*$ has less than $\lceil |U^*|/2 \rceil$ neighbors, then removing v from H^* will result in a heavier subgraph. The proof for $u \in U^*$ is symmetric. ■

Lemma 8.3 *Let $H^* = (U^*, V^*, E^*)$ be a maximum weight subgraph of G. For each set $X \subseteq U^*$ there exists a subset $Y \subseteq X$ with $|Y| \geq \lceil |X|/2 \rceil$ such that $Y \subseteq N(v)$ for some $v \in V^*$.*

Proof: Assume there exists $X \subseteq U^*$ such that all subsets $X \cap N(v), v \in V^*$ are of size smaller than $\lceil |X|/2 \rceil$. Then the weight of the subgraph induced on $(U^* \setminus X, V^*)$ exceeds that of H^*, a contradiction. ■

Corollary 8.4 *A maximum weight subgraph of G has at most $2d$ vertices from U.*

Proof: let X be U^*, since $Y \subseteq N(v)$ then $d \geq |Y|$. From the lemma we know $|Y| \geq \lceil |U^*|/2 \rceil$, so we get $U^* \leq 2d$. ■

Corollary 8.5 *Let $H^* = (U^*, V^*, E^*)$ be a maximum weight subgraph of G. Then U^* can be covered by at most $\lfloor \log(2d) \rfloor$ sets, each of which is contained in the neighborhood of some vertex in V^*.*

Proof: Denote $|U^*| = t$. By Lemma 8.3 there exists a subset $Y \subseteq U^*$ with $|Y| \geq \lceil t/2 \rceil$, such that $Y \subseteq N(v)$ for some $v \in V^*$. The same holds for the set $U^* \setminus Y$, and we can continue in this manner until we cover U^*. By construction we have at most $\lfloor \log t \rfloor$ sets in the cover. Since $t \leq 2d$ by Corollary 8.4, the result follows. ■

Corollary 8.5 implies an algorithm to find a maximum weight subgraph. The algorithm tests all collections of at most $\lfloor \log(2d) \rfloor$ subsets of neighborhoods of vertices in V. Since there are $O(n2^d)$ such subsets we have:

Theorem 8.6 *The maximum bounded bipartite subgraph problem can be solved in* $O((n2^d)^{\log(2d)})$ *time.*

8.4.5 Incorporating the Direction of Expression Changes

In our discussion so far, the underlying bipartite graph used for modeling the data contained edges for significantly changed genes, but ignored the type of change (increase or decrease in the expression level). We can integrate additional information into the model by associating a sign of "up" or "down" with each edge. We now have three types of binary relations in our bipartite graphs: An "up" edge, a "down" edge or no edge. It is reasonable to look for a bicluster in which the conditions tend to affect genes in a *consistent* way, i.e., two clustered conditions should either have always the same effect or always the opposite effect on each of the genes. This leads to the definition of a consistent biclique: Given a bipartite graph $G = (U, V, E)$ with edge sign function $c : E \to \{-1, 1\}$, we say that an induced biclique $H = (U', V', E')$ is *consistent* if there exists an assignment $\tau : U' \cup V' \to \{-1, 1\}$ such that for every $v \in V', u \in U'$ we have $c((u, v)) = \tau(u)\tau(v)$. The maximum consistent biclique problem in degree-bounded graphs can be solved in polynomial time by reduction to the standard maximum biclique problem:

Lemma 8.7 *There is an* $O(n2^d)$*-time algorithm for the maximum consistent bounded biclique problem on graphs with* d*-bounded gene side.*

Proof: Given G and c, we construct the graph $G' = (U \cup \overline{U}, V \cup \overline{V}, E')$, where $\overline{U}$ and $\overline{V}$ are copies of U and V, respectively, and $E' = \{(u, v), (\overline{u}, \overline{v}) | (u, v) \in E, c((u, v)) = 1\} \cup \{(u, \overline{v}), (\overline{u}, v) | (u, v) \in E, c((u, v)) = -1\}$. Suppose that (U', V') induce a consistent biclique in G of size k with a sign assignment τ. Then $\{v \in U' \cup V' | \tau(v) = 1\} \cup \{\overline{v} | v \in U' \cup V', \tau(v) = -1\}$ induce a biclique of size k in G'. Conversely, if (U', V') induce a biclique in G', then no pair $u, \overline{u}$ is contained in it, so $\{v \in U \cup V | v \in U' \cup V' \text{ or } \overline{v} \in U' \cup V'\}$ induce a consistent biclique in G of the same size, where $\tau(v) = 1$ if $v \in U' \cup V'$ and $\tau(v) = -1$ if $\overline{v} \in U' \cup V'$. The claim thus follows from Theorem 8.1. ■

We can use similar ideas to handle consistent subgraphs. All we need is to ensure that the sum of weights of edges from a node to two opposite sign neighbors is negative. Given this condition, the algorithms from previous sections can be applied directly and generate consistent subgraphs.

8.4.6 The SAMBA Algorithm

The scoring scheme and combinatorial algorithms developed above can be combined to create the practical SAMBA algorithm in Figure 8.4. We cannot apply the theoretical algorithm directly on large data sets since for reasonable degree bounds (e.g. 60), 2^d is too large. The suggested solution avoids hashing all subsets of incoming neighbors per gene and exhaust only subset of small size (typically $4-6$ conditions). The heaviest condition sets are then used as kernels that are extended by a local search procedure.

SAMBA thus works as follows: We first form the bipartite graph and calculate vertex pair weights using one of the weighting methods described above. We consider a gene to be up (down) regulated in a condition if its standardized level with mean 0 and variance 1 is above 1 (below -1). Depending on the data, we may choose to work with signed or unsigned graphs.

In the second phase of the algorithm we apply the hashing technique of the algorithm in Figure 8.3 to find the heaviest bicliques in the graph. In fact, SAMBA looks for the k best bicliques intersecting every given condition or gene. We ignore genes with degree exceeding some threshold d, and hash for each gene only subsets of its neighbors of size ranging from N_1 to N_2.

The third phase of the algorithm performs a local improvement procedure on the biclusters derived from the previous phase. The procedure iteratively applies the best modification to the bicluster (addition or deletion of a single vertex) until no score improvement is possible. To avoid similar biclusters whose vertex sets differ only slightly, a greedy algorithm is applied. We iterate over all generated biclusters, ordered by their score, and filter out biclusters whose intersection with a previous solution (number of shared conditions times number of shared genes) is above $L\%$.

An implementation of SAMBA can handle large data sets in a few minutes (15,000 genes, 500 conditions $d = 40, N_1 = 4, N_2 = 6, K = 20, L = 30$).

8.4.7 Validating Biclustering Quality

As the reader may have noted, for many applications in computational biology, it is hard to compare different algorithms and methodologies and state clearly which one is "better". It is however, very important for any scientific discipline to have means for evaluating the quality of a given result and to make sure the field is indeed making progress. We present examples of two general methodologies for assessing algorithms performance : comparative analysis,

SAMBA(U, V, E, w, d, N_1, N_2, k):
U : conditions. V : genes.
E : graph edges. w : edge/non-edge weights.
N_1, N_2 : hashed set size limits. k : max biclusters per gene/condition.
Initialize a hash table *weight*.
For all $v \in V$ with $|N(v)| \leq d$ do
    For all $S \subseteq N(v)$ with $N_1 \leq |S| \leq N_2$ do
        $weight[S] \leftarrow weight[S] + w(S, \{v\})$.
For each $v \in V$ set $best[v][1 \ldots k]$ to the k heaviest S such that $v \in S$.
For each $v \in V, i \in \{1..k\}$
    $S = best[v][i]$
    $V' \leftarrow \cap_{u \in S} N(u)$.
    $B \leftarrow S \cup V'$.
    Do {
        $a = argmax_{x \in V \cup U}(w(B \cup x))$
        $b = argmax_{x \in B}(w(B - x))$
        if $w(B \cup a) > w(B - b)$ then $B = B \cup a$
        else $B = B - b$
    } **While** improving
    Store B.
Post process to filter overlapping biclusters.

Figure 8.4: The practical biclustering algorithm.

which matches algorithmic results with some external knowledge, and intrinsic validation, which uses randomization to evaluate the significance of the signals discovered.

Comparative analysis

One way for evaluating bicluster algorithms is by using prior biological knowledge as some form of a gold standard that we can compare to our set of biclusters. A *correspondence plot* depicts the distribution of p-values of the produced biclusters, using for evaluation a known (putatively correct) classification of conditions (e.g., to various cancer types) or a given gene annotation. We describe the plot when a classification is given. For each value of p on a logarithmic scale, the plot presents the fraction of biclusters whose p-value is at most p out of the (say) 100 best biclusters.

p-values are calculated according to the known classification as follows: Suppose prior knowledge partitions the m conditions into k classes, $C_1, \ldots, C_k$. Let B be a bicluster with b conditions, out of which b_j belong to class C_j. The p-value of B, assuming its most abundant

class is C_i, is calculated as $p(B) = \sum_{k=b_i}^{b} \binom{|C_i|}{k}\binom{m-|C_i|}{b-k}/\binom{m}{b}$. Hence, the p-value measures the probability of obtaining at least b_i elements from the class in a random set of size b. One should note, that high quality biclusters can also identify phenomena that are not covered by the given classification. Nevertheless, we expect a large fraction of the biclusters to conform to the known classification.

As an example for the usage of correspondence plots, we present the analysis of outputs from SAMBA algorithm compared to Cheng and Church's algorithm. Running on the same data set (the lymphoma data of [2]), a collection of biclusters from both algorithm were analyzed using the known classification of conditions to different clinical types (DLBCL, CLL, FL and more). The results (Figure 8.5(A)) clearly indicate that SAMBA's biclusters are much more aligned with the biological information.

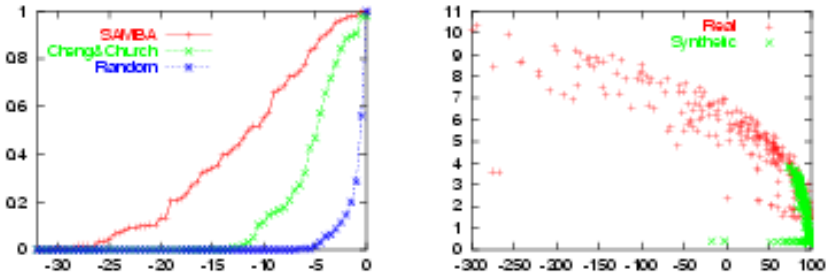
Intrinsic validation

A second, important method for validating the quality of biclusters is by analyzing the results on random data sets. We should make sure that the results we consider as statistically significant are not obtained from random data. The details of randomization may be critical to the integrity of such test. Assume we are using a uniformly random graph model (the simple model described above) and we randomize the data according to it. Then the artifact we have mentioned before, causing the identification of high degree nodes as biclusters would not be discovered since the random graph model will follow our originally restricting assumption.

Figure 8.5(B) describes the results of a randomization test done on SAMBA using a degree preserving random graph model. The analysis was done on two data sets, first the real data was biclustered and the significance of each bicluster was calculated. Then the same procedure was done with a random graph preserving all vertex degrees. The scatter plot not only demonstrate that heavy biclusters are non random but also provides empirical evidence to the relation of the SAMBA likelihood score and the more formal significance measure.

8.4.8 Functional Annotation using biclusters

The output of a bicluster algorithm is a collection of significant local signals in the data. Such signals may be used in diverse application. One example for an application of biclusters is for automatic annotation of genes. The budding yeast, which is one of the most characterized eukaryotes up to date, contains about 6200 genes, only half of which have a known function. We can use a large database of gene expression and a set of derived biclusters to try and associate unknown genes with some function. The idea is simple, whenever a majority of the characterized genes in a bicluster share a common functional class, it is likely that the other genes in the bicluster are also related to this class.



A B

Figure 8.5: Source: [15]. A: Performance of different weighting schemes and algorithms. Correspondence plots for SAMBA, the algorithm of Cheng and Church [4], and random biclusters. Likelihood weights use $p_c = 0.9$. B: Scatter plots of significance values on synthetic and real data. x-axis: significance value, y-axis: bicluster weight.

A compiled data set of yeast gene expression, including 515 conditions for the 6,200 yeast ORFs was used to test this idea. The data was collected from five different experiments [10, 6, 7, 14, 11]. Analysis by SAMBA generated 2,406 biclusters ranging over 4,961 genes and 515 conditions. The source for the known functional annotation was the SGD database [1], which includes 3000 annotations using the Gene Ontology [5] vocabulary.

The bicluster set was filtered to include only those biclusters in which more than 60% of their annotated members had the same class. Out of those, only biclusters that were functionally enriched (p-value below 10^{-4}) were used. The unannotated genes in those biclusters were now assigned to the most abundant class. Note that each gene may be annotated more than once, as is the case for the curated GO annotations. For cross validation, 100 runs were performed and in each one 30% of the annotations were hidden. The overall average success rate in annotating the hidden genes was calculated.

The results of these runs are summarized in Figure 8.6(A,B). Overall, 81.5% of the test set annotations matched those known from SGD, demonstrating that we can extrapolate functional annotation using biclusters.

8.5 Summery

In this and the previous lectures, we have reviewed some methods for biclustering. In each of these methods we focused on the discovery of one, presumingly best, bicluster. To obtain a set of biclusters we had to apply our algorithm many times, possibly in parallel. A different approach may be to try and discover a multiple biclustering model for the entire data. This approach has the advantage of enabling a global view of the data and reducing overfitting and redundancies. On the other hand, given that bicluster are by definition a local pattern,

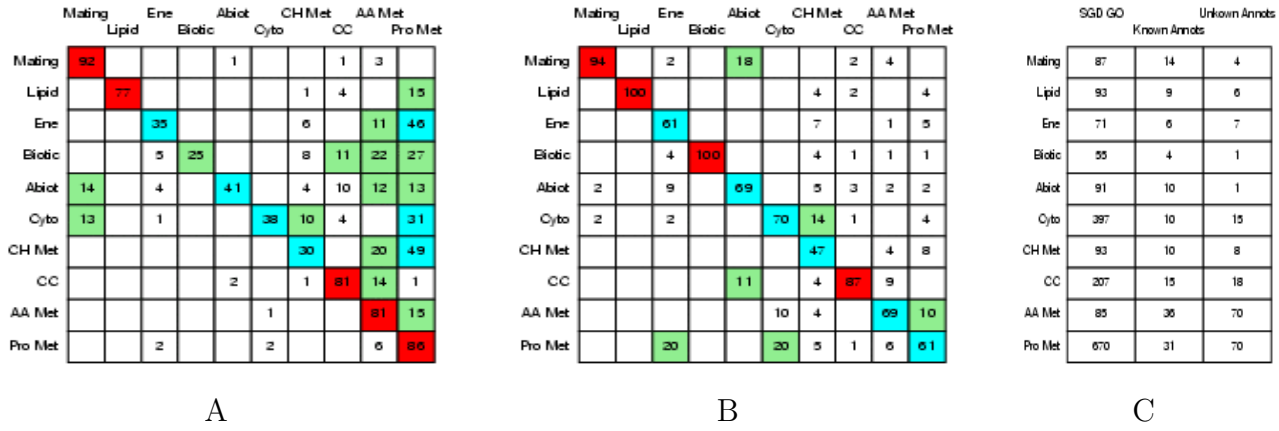


Figure 8.6: Source: [15]. Yeast functional annotation. A: Annotation specificity. The table depicts the annotation accuracy measured using 70:30 cross-validation. Rows represent classes assigned using our method and columns represent SGD GO classes. Cell (x, y) contains the percentage of genes annotated x that belong to GO class y. Higher percentages are darker. B: Annotation sensitivity calculated w.r.t. annotated genes only. Cell (x, y) contains the percentage of SAMBA annotated genes that belong to GO class y and were annotated x. C: Annotation of unknown genes. The table shows for each functional class its size in the SGD GO annotation, the number of genes that belong to this class and were annotated by SAMBA, and the number of unknown genes assigned to this class by SAMBA. Abbreviations for functional classes: Mating - mating (sensu Saccharomyces, Fungi); Lipid - lipid metabolism; Ene - energy pathway; Biotic - response to biotic stimulus; Abiot - response to abiotic stimulus; Cyb - cytoplasm organization and biogenesis; CH Met - carbohydrate metabolism; CC - mitotic cell cycle; AA Met - amino acid and derivative metabolism; Pro Met - protein metabolism and modification.

a global approach to biclustering is not a trivial task. A family of methods using a basically global approach is based on probabilistic models. For example, we can model each sub column and raw as having a typical value distribution, which may be different from the background. We can then model the entire data set by tiling it with a small number of biclusters, each having a characteristic distribution. To discover such tiling, it is possible to use variants of the EM algorithm.

As we have seen, the many possible definition of the biclustering problem have a central importance in the design and implementation of algorithms to solve it. As of now, there is no one single solution that had been accepted as a standard. As gene expression data accumulates rapidly, the need for a robust and efficient biclustering algorithms is nevertheless increasing.

Bibliography

[1] http://genome-www.stanford.edu/saccharomyces/.

[2] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. Hudson, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chan, T. C. Greiner, D. D. Weisenburger, J. O. Armitage, R. Warnke, and L. M. Staudt. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403(6769):503–511, 2000.

[3] S. Bergmann, J. Ihmels, and N. Barkai. Iterative signature algorithm for the analysis of large-scale gene expression data. *Phys Rev E Stat Nonlin Soft Matter Phys*, 67(3 Pt 1):03190201–18, 2003.

[4] Y. Cheng and G.M. Church. Biclustering of expression data. In *Proc. ISMB'00*, pages 93–103. AAAI Press, 2000.

[5] The Gene Ontology Consortium. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[6] A. P. Gasch et al. Genomic expression programs in the response of yeast cells to environmental changes. *Mol Biol Cell*, 11:4241–57, 2000.

[7] A.P. Gasch et al. Genomic expression responses to DNA-damaging agents and the regulatory role of the yeast ATR homolog mec1p. *Mol. Biol. Cell*, 12(10):2987–3003, 2001.

[8] G. Getz, E. Levine, and E. Domany. Coupled two-way clustering analysis of gene microarray data. *Proc. Natl. Acad. Sci. USA*, 97(22):12079–84, 2000.

[9] Dorit S. Hochbaum. Approximating clique and biclique problems. *Journal of Algorithms*, 29(1):174–200, 1998.

[10] TR. Hughes et al. Functional discovery via a compendium of expression profiles. *Cell*, 102:109–26, 2000.

[11] T. Ideker, J.A. Thorsson, V. Ranish, R. Christmas, J. Buhler, J.K. Eng, R. Bumgarner, D.R. Goodlett, R. Aebersold, and Hood L. Integrated genomic and proteomic analyses of a systematically perturbed metabolic network. *Science*, 291:929–34, 2001.

[12] J. Ihmels, G. Friedlander, S. Bergmann, O. Sarig, Y. Ziv, and N. Barkai. Revealing modular organization in the yeast transcriptional network. *Nature Genetics*, 31(4):370–7, 2002.

[13] R. Peeters. The maximum edge biclique problem is NP-complete. citeseer.nj.nec.com/peeters00maximum.html.

[14] P. T. Spellman, G. Sherlock, et al. Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol. Biol. Cell*, 9:3273–3297, 1998.

[15] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18(Suppl 1):S136–44, 2002.