Introduction to Biclustering

SAMBA Algorithm

Presents: Stefan Gremalschi

CSc: 8910
Summary

- **SAMBA (Statistical Algorithmic Method for Bicluster Analysis)**
  - Data Modeling
    - A Simple Model
    - A Refined Model
  - Finding Heavy Subgraphs
    - Maximum Bounded Biclique
    - Finding Heavy Subgraphs
  - Incorporating the Direction of Expression Changes
  - Significance Evaluation
  - The Samba Algorithm
  - Validating Biclustering Quality
    - Comparative Analysis
    - Intrinsic Validation
  - Functional Annotation using Biclusters
Data Modeling

- 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 with respect to 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. We shall assign weights to the vertex pairs of the bipartite graph according to a statistical model. 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.

- Given an input gene expression data, we form a bipartite graph $G = (U, V, E)$. $U$ is the set of conditions, $V$ is the set of genes, and $(u, v)$ belongs to $E$ iff $v$ responds in condition $u$, that is, if the expression level of $v$ changes significantly in $u$.

- 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 $U'$. The weight of a subgraph (bicluster) is the sum of the weights of gene-condition pairs in it, including edges and non-edges.
Data Modeling

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. This is done so that the score can be expressed as a sum of independent contributions from each of the node pairs in the subgraph.

Using this model, we can reduce the biclustering problem to the problem of finding heavy subgraphs in a bipartite graph.

**Figure:** Gene expression data is modeled using a bipartite graph with two sides U and 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 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 $H = (U^1, V^1, E^1)$ be a subgraph of $G$.
- Denote $|U^1| = m^1$, $|V^1| = n^1$, $|E^1| = k^1$. The model assumes that edges occur independently with equal probably $p$, where $p = |E| / (|U| |V|)$ (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

$$P(H) = BT(k', p, n'm') = \sum_{k''=k'}^{n'm'} \binom{n'm'}{k''} p^{k''} (1-p)^{n'm' - k''}$$

- The goal is to find a subgraph $H$ with the lowest $P(H)$. By bounding the terms of the binomial tail using the first term (the largest, assuming that $p < 1/2$), we obtain the following upper bound for $P(H)$:

$$P^*(H) \leq p^{k'} (1-p)^{n'm' - k'} \sum_{k''=k'}^{n'm'} \binom{n'm'}{k''} \leq 2^{n'm'} p^{k'} (1-p)^{n'm' - k'}$$

- A subgraph $H$ minimizing log $P^*(H)$ is equivalent to 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.
A Refined Model

• The Simple Model is far from the reality. The degree distribution of real gene expression data has a very non uniform behavior, where some of the conditions and genes have very high degrees and others very low degrees. A simple random graph model would result in very high scores for the ”bicluster” defined by all high degree nodes in the graph.

• The **refined null model** takes into account the variability of the degrees in G. It incorporates the characteristic behavior of each specific condition and gene.
A Refined Model

- Let \( H = (U^1, V^1, E^1) \) be a subgraph of \( G \) and denote \( E^1 = (U^1 \times V^1) \setminus E^1 \). For a vertex \( w \) belonging to \( U^1 \) union \( V^1 \) let \( d^G_w \) 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

\[
p_{(u,v)} = \frac{\left| \{ G' = (U, V, E') \mid \forall w, d^G_w = d'^G_w, (u, v) \in E' \} \right|}{\left| \{ G' = (U, V, E') \mid \forall w, d'^G_w = d^G_w \} \right|}
\]

- 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. The probability of observing \( H \) is thus

\[
P(H) = \left( \prod_{(u,v) \in E^1} p_{(u,v)} \right) \cdot \left( \prod_{(u,v) \in E^c} (1 - p_{(u,v)}) \right)
\]

- 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.
Alternative Model

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:

\[
\log L(H) = \sum_{(u,v) \in E'} \log \frac{p_c}{P(u,v)} + \sum_{(u,v) \in E'} \log \frac{1 - p_c}{1 - P(u,v)}
\]

Setting the weight of each edge \((u,v)\) to \(\log (p_c / P(u,v)) > 0\) and the weight of each non-edge \((u,v)\) to \(\log (1 - p_c / 1 - P(u,v)) < 0\), we conclude that the score of H is simply its weight.
Finding Heavy Subgraphs

- The problem of identifying the maximum likelihood bicluster is finding the heaviest subgraph.
- The computational problem of finding the largest node biclique in a bipartite graph has an elegant polynomial time algorithm (using matching).
- Our problem, however, is 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 and weighted graphs.
- We enforce additional limitation on our graph: restrict the in degree of the genes side, so that a gene that respond in more than d conditions is ignored.
- We present a polynomial algorithm which is used as the basis for a practical implementation that can avoid the degree restriction. 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 $v$ from $V$ has degree at most $d$. Let $w : U \times V \rightarrow \mathbb{R}$ be a weight function. For a pair of subsets $U^1$ from $U$, $V^1$ from $V$ we denote by $w(U^1, V^1)$ the weight of the subgraph induced on $U^1$ union $V^1$, i.e.,

$$w(U^1, V^1) = \sum_{u \text{ belongs } U^1, v \text{ belongs } V^1} w((u, v))$$

• The neighborhood of a vertex $v$, denoted $N(v)$, is the set of vertices adjacent to $v$ in $G$. We denote $n = |V|$ throughout.
Maximum Bounded Biclique Algorithm

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} = \cap_{u \in U_{best}} N(u)
Output (U_{best}, V_{best})
Finding Heavy Subgraphs
Algorithm

• We will 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 \(w : U \times V \rightarrow \{-1, 1\}\) such that \(w((u,v))=1\) for \((u, v)\) belongs to \(E\), and \(w((u, v)) = -1\) for \((u, v)\) belongs to \((U \times V) \setminus E\).

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

• The maximum bounded bipartite subgraph problem can be solved in \(O((n2^d)^{\log(2d)})\) time.
Incorporating the Direction of Expression Changes

- 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.
- Two clustered conditions should either have always the same effect or always the opposite effect on each of the genes.
- 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 \( H=(U^1,V^1,E^1) \) is consistent if there exists an assignment: \( U^1 \cup V^1 \to \{-1,1\} \) such that for every \( v \in V^1 \), \( u \in U^1 \) 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. There is an \( O(n2^d) \)-time algorithm for the maximum consistent bounded biclique problem on graphs with \( d \)-bounded gene side.
The SAMBA Algorithm

SAMBA 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 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 N1 to N2.

• 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, N1 = 4, N2 = 6, K = 20, L = 30).
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 $v \in V, i \in \{1 \ldots k\}$
  $S = best[v][i]$
  $V' \leftarrow \cap_{u \in S} N(u)$.
  $B \leftarrow S \cup V'$.
  Do {
    $a = \text{argmax}_{x \in V \cup U} (w(B \cup x))$
    $b = \text{argmax}_{x \in B} (w(B - x))$
    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.
Validating Biclustering Quality

- 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**, 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 for evaluating bicluster algorithms is by using prior biological knowledge as some form of a gold standard. For each value of p on a logarithmic scale, the plot (next slide) presents the fraction of biclusters whose p-value is at most p out of the 100 best biclusters.

- p-values are calculated according to the known classification as follows: prior knowledge partitions the m conditions into k classes, $C_1, ..., C_k$. 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

$$p(B) = \sum_{k=b_i}^{b} \binom{|C_i|}{k} \binom{m-|C_i|}{b-k} / \binom{m}{b}$$

- The p-value measures the probability of obtaining at least $b_i$ elements from the class in a random set of size b. We expect a large fraction of the biclusters to conform to the known classification.

- Usage of correspondence plots: the analysis of outputs from SAMBA algorithm compared to Cheng and Church’s algorithm. Running on the same data set (the lymphoma data), a collection of biclusters from both algorithm were analyzed. The results 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 and we randomize the data according to it. Then the artifact causing the identification of high degree nodes as biclusters would not be discovered since the random graph model will follow our originally restricting assumption.
- The plot 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 and 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.
• **Figure**: 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.
An application of biclusters is automatic annotation of genes. 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 compiled data set of yeast gene expression, including 515 conditions for the 6,200 yeast ORFs was used to test this idea. 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, which includes 3000 annotations using the Gene Ontology 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. 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. Overall, 81.5% of the test set annotations matched those known from SGD, demonstrating that we can extrapolate functional annotation using biclusters.
**Figure.** Yeast functional annotation. A: Annotation specificity. The table depicts the annotation accuracy measured using 70:30 cross-validation. 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.
Fig. 5. Sample biclusters. Each figure shows the expression patterns in two related biclusters. Rows correspond to genes and columns correspond to conditions or tissues. Expression levels: Dark – up; light – down; black – unchanged. The frames indicate bicluster boundaries.

(a) Yeast biclusters. A group of unannotated subtelomeric Y' genes is clustered with several DNA repair (DR) genes (upper left corner). This raises the hypothesis of association between DNA repair mechanisms and the Y' genes, which was independently suggested recently. Some of the genes in this bicluster appear also in another presented at the lower right corner, which contains phosphate (P) and glucose (G) related genes. Several unannotated genes (Un) may be assigned a putative function in this way. (b) Biclusters in lymphoma data. Germinal center (GC) tissues are biclustered with both DLBCL and FL tissues, thus uniquely characterizing them as a distinct class.

Detailed analysis of the results demonstrates the power of bicluster analysis. For example, one of the biclusters in Figure 5a contains DNA repair genes and a large family of Y' DNA helicase genes. The Y' genes are strong paralogs present at the end of the yeast chromosome, and their function is not fully understood. This bicluster raises the conjecture that Y' genes and DNA repair genes are associated. Indeed, a recent study (Yamada et al., 1998) suggested a connection between DNA damage and repair mechanisms to this family. Another bicluster shown in this figure contains several phosphate and glucose related genes grouped with several unknown genes, which may be assigned a putative function according to their expression pattern in this bicluster.
Questions