Computing Bi-Clusters for Microarray Analysis

Yu Lin

December 21, 2006
General Bi-clustering Problem

- **Input**: a $n \times m$ matrix $A$.
- **Output**: a sub-matrix $A_{P,Q}$ of $A$ such that the rows of $A_{P,Q}$ are *similar*. That is, all the rows are identical.

Why sub-matrix?
A subset of *genes* are co-regulated and co-expressed under specific *conditions*. It is interesting to find the subsets of genes and conditions.
Similarity of Rows (1-5)

1. All rows are identical
   1 1 2 3 2 3 3 2
   1 1 2 3 2 3 3 2
   1 1 2 3 2 3 3 2

2. All the elements in a row are identical
   1 1 1 1 1 1 1 1
   2 2 2 2 2 2 2 2
   5 5 5 5 5 5 5 5
   (the same as 1 if we treat columns as rows)
Similarity of Rows (1-5)

3. The curves for all rows are similar (additive)
   \[ a_{i,j} - a_{i,k} = c(j, k) \] for \( i = 1, 2, \ldots, m \). Case 3 is equivalent to case 2 (thus also case 1) if we construct a new matrix \( a_{i,j}^* = a_{i,j} - a_{i,p} \) for a fixed \( p \) indicate a row.
Similarity of Rows (1-5)

4. The curves for all rows are similar (multiplicative)

\[
\begin{align*}
    a_{1,1} & \quad a_{1,2} & \quad a_{1,3} & \quad \ldots & \quad a_{1,m} \\
    c_{1}a_{1,1} & \quad c_{1}a_{1,2} & \quad c_{1}a_{1,3} & \quad \ldots & \quad c_{1}a_{1,m} \\
    c_{2}a_{1,1} & \quad c_{2}a_{1,2} & \quad c_{2}a_{1,3} & \quad \ldots & \quad c_{2}a_{1,m} \\
    & \quad \ldots & \quad \ldots & \quad \ldots & \quad \ldots \\
    c_{n}a_{1,1} & \quad c_{n}a_{1,2} & \quad c_{n}a_{1,3} & \quad \ldots & \quad c_{n}a_{1,m}
\end{align*}
\]

Transfer to case 2 (thus case 1) by taking log and subtraction. Case 3 and Case 4 are called bi-clusters with coherent values.
Similarity of Rows (1-5)

5. The curves for all rows are similar (multiplicative and additive)

\[ a_{i,j} = c_i a_{k,j} + d_i \]

Transfer to case 2 (thus case 1) by subtraction of a fixed row (row i), taking log and subtraction of row i again. The basic model: All the rows in the sub-matrix are identical.
Cheng and Church's model

The model introduced a similarity score called the mean squared residue score $H$ to measure the coherence of the rows and columns in the submatrix.

$$H(P, Q) = \frac{1}{|P||Q|} \sum_{i \in P, j \in Q} (a_{i,j} - a_{i,Q} - a_{P,j} + a_{P,Q})^2$$

where

$$a_{i,Q} = \frac{1}{|Q|} \sum_{j \in Q} a_{i,j}, \quad a_{P,j} = \frac{1}{|P|} \sum_{i \in P} a_{i,j}, \quad a_{P,Q} = \frac{1}{|P||Q|} \sum_{i \in P, j \in Q} a_{i,j}.$$

If there is no error, $H(P, Q) = 0$ for case 1, 2 and 3. A lot of heuristics (programs) have been produced.
Our Problem Definition

- Consensus Sub-matrix Problem
- Bottleneck Sub-matrix Problem
Consensus Sub-matrix Problem

- Input: a $n \times m$ matrix $A$, integers $l$ and $k$.
- Output: a sub-matrix $A_{P,Q}$ of $A$ with $l$ rows and $k$ columns and a consensus row $z$ (of $k$ elements) such that

$$\sum_{r_i \in P} d(r_i|Q, z)$$

is minimized.

Here $d(,)$ is the Hamming distance.
Bottleneck Sub-matrix Problem

- Input: a $n \times m$ matrix $A$, integers $l$ and $k$.
- Output: a sub-matrix $A_{P,Q}$ of $A$ with $l$ rows and $k$ columns and a consensus row $z$ (of $k$ elements) such that for any $r_i$ in $P$

$$d(r_i|Q,z) \leq d$$

and $d$ is minimized

Here $d(\ ,\ )$ is the Hamming distance.
NP-Hardness Results

- Theorem 1: Both consensus sub-matrix and bottleneck sub-matrix problems are NP-hard.

Proof: We use a reduction from maximum edge bipartite problem.
Approximation Algorithm for Consensus Sub-matrix Problem

- **Input**: a $n \times m$ matrix $A$, integers $l$ and $k$.
- **Output**: a sub-matrix $A_P, Q$ of $A$ with $l$ rows and $k$ columns and a consensus row $z$ (of $k$ elements) such that

\[
\sum_{r_i \in P} d(r_i|Q, z) \text{ is minimized.}
\]

Here $d(\ , \ )$ is the Hamming distance.
Basic Ideas

Assumptions: \( H_{opt} = \sum_{p_i \in P_{opt}} d(x_{p_i} | Q_{opt}, z_{opt}) = O(kl), \quad H_{opt} \times c' = kl \)
and \( |Q_{opt}| = k = O(n), \quad k \times c = n. \)

Basic Ideas: We use a random sampling technique to randomly select \( O(\log m) \) columns in \( Q_{opt} \), enumerate all possible vectors of length \( O(\log m) \) for those columns. At some moment, we know \( O(\log m) \) bits of \( r_{opt} \) and we can use the partial \( z_{opt} \) to select the \( l \) rows which are closest to \( z_{opt} \) in those \( O(\log m) \) bits. After that we can construct a consensus vector \( r \) as follows: for each column, choose the (majority) letter that appears the most in each of the \( l \) letters in the \( l \) selected rows. Then for each of the \( n \) columns, we can calculate the number of mismatches between the majority letter and the \( l \) letters in the \( l \) selected rows. By selecting the best \( k \) columns, we can get a good solution.
Basic Ideas

- How to randomly select $O(\log m)$ columns in $Q_{opt}$ while $Q_{opt}$ is unknown?
- Our new idea is to randomly select a set $B$ of $(c + 1)\log m$ columns from $A$ and enumerate all size $\log m$ subsets of $B$ in time $O(m^{c+1})$ which is polynomial in terms of the input size $O(mn)$. We can show that with high probability, we can get a set of $\log m$ columns randomly selected from $Q_{opt}$.
Algorithm 1 for The Consensus Submatrix Problem

Input: one $m \times n$ matrix $A$, integers $l$ and $k$, and $\epsilon > 0$

Output: a size $l$ subset $P$ of rows, a size $k$ subset $Q$ of columns and a length $k$ consensus vector $z$

Step 1: randomly select a set $B$ of $\lceil (c + 1)(\frac{4 \log m}{\epsilon^2} + 1) \rceil$ columns from $A$.

(1.1) for every size $\lceil \frac{4 \log m}{\epsilon^2} \rceil$ subset $R$ of $B$ do

(1.2) for every $z^{|R|} \in \Sigma^{|R|}$ do

(a) Select the best $l$ rows $P = \{p_1, ..., p_l\}$ that minimize $d(z^{|R|}, x_i^{|R|})$.

(b) for each column $j$ do

Compute $f(j) = \sum_{i=1}^{l} d(s_j, a_{p_i,j})$, where $s_j$ is the majority element of the $l$ rows in $P$ in column $j$. Select the best $k$ columns $Q = \{q_1, ..., q_k\}$ with minimum value $f(j)$ and let $z(Q) = s_{q_1}s_{q_2}...s_{q_k}$.

(c) Calculate $H = \sum_{i=1}^{l} d(x_{p_i}^{|Q|}, z)$ of this solution.

Step 2: Output $P$, $Q$ and $z$ with minimum $H$. 
Proofs

► Lemma 1: With probability at most \( m - \frac{2}{\epsilon^2 c^2 (c+1)} \), no subset \( R \) of size \( \left\lceil \frac{4 \log m}{\epsilon^2} \right\rceil \) used in Step 1 of Algorithm 1 satisfies \( R \subseteq Q_{opt} \).

► Lemma 2: Assume \( |R| = \left\lceil \frac{4 \log m}{\epsilon^2} \right\rceil \) and \( R \subseteq Q_{opt} \). Let \( \rho = \frac{k}{|R|} \). With probability at most \( m^{-1} \), there is a row \( x_i \) in \( X \) satisfying

\[
\frac{d(z_{opt} | Q_{opt}), x_i | Q_{opt}) - \epsilon k}{\rho} > d(z_{opt} | R), x_i | R).
\]

With probability at most \( m^{-\frac{1}{3}} \), there is a row \( x_i \) in \( X \) satisfying

\[
d(z_{opt} | R), x_i | R) > \frac{d(z_{opt} | Q_{opt}), x_i | Q_{opt}) + \epsilon k}{\rho}.
\]
Proofs

- Lemma 3: When $R \subseteq Q_{opt}$ and $z^R = z_{opt}^R$, with probability at most $2m^{-\frac{1}{3}}$, the set of rows $P = \{p_1, \ldots, p_l\}$ selected in Step 1 (a) of Algorithm 1 satisfies
  \[ \sum_{i=1}^l d(z_{opt}, x_{p_i}|Q_{opt}) > H_{opt} + 2\epsilon kl. \]

- Theorem 2: For any $\delta > 0$, with probability at least
  \[ 1 - m^{-\frac{8c'}{\delta^2c'^2(c+1)}} - 2m^{-\frac{1}{3}}, \]
  Algorithm 1 will output a solution with consensus score at most $(1 + \delta)H_{opt}$ in $O(nm^{O(\frac{1}{\delta^2})})$ time.
Approximation Algorithm for Bottleneck Sub-matrix Problem

- Input: a $n \times m$ matrix $A$, integers $l$ and $k$.
- Output: a sub-matrix $A_{P,Q}$ of $A$ with $l$ rows and $k$ columns and a consensus row $z$ (of $k$ elements) such that for any $r_i$ in $P$

$$d(r_i|_Q, z) \leq d$$

and $d$ is minimized

Here $d(\ , \ )$ is the Hamming distance.
Basic Ideas

▶ Assumptions: $d_{opt} = MAX_{p_i \in P_{opt}} d(x_{p_i} | ^{Q_{opt}}, z_{opt}) = O(k)$, $d_{opt} \times c'' = k$ and $|Q_{opt}| = k = O(n)$, $k \times c = n$.

▶ Basic Ideas:
(1) Use random sampling technique to know $O(\log m)$ bits of $z_{opt}$ and select $l$ best rows like Algorithm 1.
(2) Use linear programming and randomized rounding technique to select $k$ columns in the matrix.
Linear programming
Given a set of rows $P = \{p_1, \ldots, p_l\}$, we want to find a set of $k$ columns $Q$ and vector $z$ such that bottleneck score is minimized.

$$\min d;$$
$$\sum_{i=1}^{n} \sum_{j=1}^{\left|\Sigma\right|} y_{i,j} = k,$$
$$\sum_{j=1}^{\left|\Sigma\right|} y_{i,j} \leq 1, i = 1, 2, \ldots, n,$$
$$\sum_{i=1}^{n} \sum_{j=1}^{\left|\Sigma\right|} \chi(\pi_j, \chi_{p_s,i})y_{i,j} \leq d, s = 1, 2, \ldots, l.$$

$y_{i,j} = 1$ if and only if column $i$ is in $Q$ and the corresponding bit in $z$ is $\pi_j$.

Here, for any $a, b \in \Sigma$, $\chi(a, b) = 0$ if $a = b$ and $\chi(a, b) = 1$ if $a \neq b$. 
Randomized rounding
To achieve two goals:
(1) Select \( k' \) columns, where \( k' \geq k - \delta d_{opt} \).
(2) Get integers values for \( y_{i,j} \) such that the distance (restricted on the \( k' \) selected columns) between any row in \( P \) and the center vector thus obtained is at most \( (1 + \gamma) d_{opt} \).

Here \( \delta > 0 \) and \( \gamma > 0 \) are two parameters used to control the errors.
Lemma 4: When \( \frac{n \gamma^2}{3(2c'')^2} \geq 2 \log m \), for any \( \gamma, \delta > 0 \), with probability at most \( \exp(-\frac{n \delta^2}{2(2c'')^2}) + m^{-1} \), the rounding result \( y' = \{ y'_{1,1}, \ldots, y'_{1,|\Sigma|}, \ldots, y'_{n,1}, \ldots, y'_{n,|\Sigma|} \} \) does not satisfy at least one of the following inequalities,

\[
\sum_{i=1}^{n} \left( \sum_{j=1}^{|\Sigma|} y'_{i,j} \right) > k - \delta d_{opt},
\]

and for every row \( x_{p_s} (s = 1, 2, \ldots, l) \),

\[
\sum_{i=1}^{n} \left( \sum_{j=1}^{|\Sigma|} \chi(\pi_j, x_{p_s,i})y'_{i,j} \right) < \overline{d} + \gamma d_{opt}.
\]
Algorithm 2 for The bottleneck Sub-matrix Problem

**Input:**
one matrix $A \in \Sigma^{m \times n}$, integer $l$, $k$, a row $z \in \Sigma^n$ and small numbers $\epsilon > 0$, $\gamma > 0$ and $\delta > 0$.

**Output:** a size $l$ subset $P$ of rows, a size $k$ subset $Q$ of columns and a length $k$ consensus vector $z$.

If $\frac{n \gamma^2}{3 (cc')^2} \leq 2 \log m$ then try all size $k$ subset $Q$ of the $n$ columns and all $z$ of length $k$ to solve the problem.

If $\frac{n \gamma^2}{3 (cc')^2} > 2 \log m$ then

**Step 1:** randomly select a set $B$ of $\lceil \frac{4(c+1) \log m}{\epsilon^2} \rceil$ columns from $A$. For every $\lceil \frac{4 \log m}{\epsilon^2} \rceil$ size subset $R$ of $B$ do

For every $z^R \in \Sigma^{|R|}$ do

(a) Select the best $l$ rows $P = \{p_1, ..., p_l\}$ that minimize $d(z^R, x_i^R)$.

(b) Solve the optimization problem by linear programming and randomized rounding to get $Q$ and $z$.

**Step 2:** Output $P, Q$ and $z$ with minimum bottleneck score $d$. 
Proofs

Lemma 5: When $R \subseteq Q_{\text{opt}}$ and $z|^R = z_{\text{opt}}|^R$, with probability at most $2m^{-\frac{1}{3}}$, the set of rows $P = \{p_1, \ldots, p_l\}$ obtained in Step 1(a) of Algorithm 2 satisfies 
$$d(z_{\text{opt}}, x_{p_i}|Q_{\text{opt}}) > d_{\text{opt}} + 2\epsilon k$$ for some row $x_{p_i}(1 \leq i \leq l)$.

Theorem 3: With probability at least 
$$1 - m^{-\frac{2}{e^2c^2(c+1)}} - 2m^{-\frac{1}{3}} - \exp\left(-\frac{m\delta^2}{2(c'c'')^2}\right) - m^{-1},$$
Algorithm 2 runs in time $O(n^{O(1)} m^{O(\frac{1}{c^2} + \frac{1}{c'^2})})$ and obtains a solution with bottleneck score at most $(1 + 2c''\epsilon + \gamma + \delta)d_{\text{opt}}$ for any fixed $\epsilon, \gamma, \delta > 0$. 
Thanks

- Acknowledgements
  This work is fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China [Project No. CityU 1070/02E].
  This work is collaborated with Dr. Lusheng Wang and Xiaowen Liu in City University of Hong Kong, Hong Kong, China.
Let $X_1, X_2, \ldots, X_n$ be $n$ independent random 0-1 variables, where $X_i$ takes 1 with probability $p_i$, $0 < p_i < 1$. Let $X = \sum_{i=1}^{n} X_i$, and $\mu = E[X]$. Then for any $0 < \epsilon \leq 1$,

$$\Pr(X > \mu + \epsilon n) < e^{-\frac{1}{3}n\epsilon^2},$$

$$\Pr(X < \mu - \epsilon n) \leq e^{-\frac{1}{2}n\epsilon^2}.$$