Lecture 15

Group Testing (An overview)

Date back: 1943, Blood testing during World War II
(Dorfman and Rosenblatt)
(First GT paper by Dorfman.)

Group Testing Problem

Classical

Given a set N of n clones (→ items (in general)), each of which is either positive or negative but not both. The goal of group testing is to identify all positive ones (positives in short) by using as few number of "group tests" as possible. Here, a group test is an experiment on a subset of clones and the outcome of a group test on a subset $S \subseteq N$ is "1" if S contains at least one positive clone and "0" otherwise.

Example $N = \{1,2,3,4,5,6\}$ and 3 is the only positive clone.
Then $t(\{1,3,3\}) = 1$ and $t(\{2,4,5\}) = 0$.

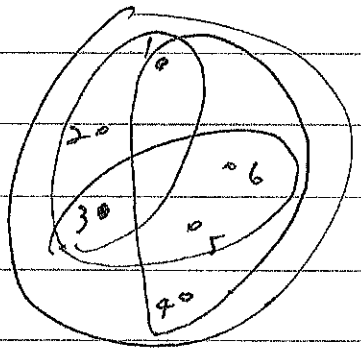
- Assumption on combinatorial group testing.
  Let d be the number of positives in a set N of size n.
  Then, we assume that $d \ll n$, i.e., d/n is very close to "0".

(Note) If d is too large comparing to n, then "group testing is not working well in general.

Algorithms

1. Adaptive : An adaptive algorithm conducts tests one by one (Sequential)
2. Non-adaptive : A non-adaptive algorithm specifies all tests simultaneously.
3. h-stage algorithm : All tests in the "first" stage are specified simultaneously.

- The non-adaptive algorithm is also known as pooling design which can be depicted by using a matrix M.
  Let the number of clones be n and there are t tests.
  The the matrix M corresponding to a pooling design has the following properties :
  (a) M is a t by n matrix ; and
  (b) $M(i,j) = \begin{cases} 1 & \text{if the } i^{th} \text{ test including clone "j" ; and} \\ 0 & \text{otherwise.} \end{cases}$



outcome vector

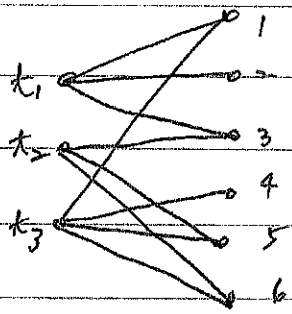
$$\begin{array}{c} \\ t_1 \\ t_2 \\ t_3 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \end{array} \begin{bmatrix} \\ \\ \end{bmatrix}$$

If (3) is positive, then the outcome vector is $(1,1,0)^t$.

How much do we know about (0,1)-matrix ?

(Fact 1) If M is a $t \times n$ (0,1)-matrix, then we can find a labeled bipartite graph whose incidence matrix is M. Here $|A| = t$ and $|B| = n$. $G = (A, B)$

So, the above matrix is corresponding to the following bipartite graph



(Fact 2) If we let A be the set of elements and B is the collection of blocks, then we have a variety-block graph.

Again, the above example shows that there are six blocks $\{\{1,3\}, \{1\}, \{1,2\}, \{3\}, \{2,3\}, \{2,3\}\}$.

- A design can be described by using a (0,1)-matrix, so is a hypergraph.

(Fact 3) A $t \times n$ (0,1)-matrix M can also be seen as a code where codewords are columns of M.

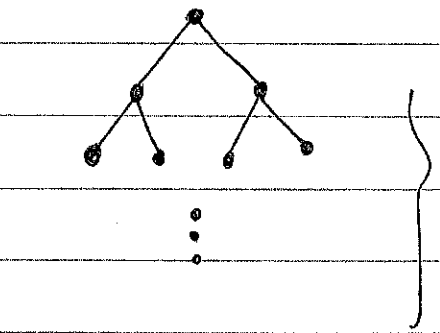
(Note) For practical use, we don't consider codewords from rows! (But, in G.T., each row is a test (pool).)

(*) The idea of finding positives

If we have d positives, then there are $\binom{n}{d}$ possible combinations to have d positives. In order to find these positives, we may use different algorithms. In general, adaptive algorithms take less tests than non-adaptive algorithm. But, the later one "save time".

Information lower bound

We can use the idea of computational tree to find how many tests are needed (at least).



$$2^h = \binom{n}{d} \Rightarrow h \approx \left\lceil \log_2 \binom{n}{d} \right\rceil .$$

(*) We need at least $\left\lceil \log_2 \binom{n}{d} \right\rceil$ tests!

Another point of view

Consider all possible outcomes if t tests are applied. Then, there are at most 2^t outcome vectors which are distinct. So, if we use t tests to find d positives out of n, then $\binom{n}{d} \le 2^t$ and thus

$$t \ge \left\lceil \log_2 \binom{n}{d} \right\rceil \approx d \log_2 \frac{n}{d}.$$

(Note) This idea is good for both adaptive and non-adaptive algorithms.

(Fact 4) With the idea of checking the outcome vectors, we may extend the study of error-correcting codes to construct pooling designs which can also solve the group testing problem with the occurrence of errors in experiments.

In what follows, we shall focus on non-adaptive algorithms or k-stage algorithms due to the fact that apply them in computational molecular biology.

Definition (Pool)

Each test is also said to be a pool. A positive pool is a test (corresponding to the pool) which is positive, i.e., some of the clones in the experiment is positive and a negative pool is defined correspondingly, i.e., every clone in this pool is negative.

Separable matrices

$M = [M(i,j)]_{m \times n}$ (t tests and n clones)

- We shall index the columns of M by $1, 2, \cdots, n$ and the rows of M by $t_1, t_2, \cdots, t_m$.
- The clone (j^{th}) can be represented by a set C_j where $C_j = \{i \mid M(i,j) = 1\}$.
- The union of s columns $C_{j_1}, C_{j_2}, \cdots, C_{j_s}$ is $\bigcup_{k=1}^{s} C_{j_k}$ which is the Boolean sum of these columns.

$\bigcup_{k=1}^{s} C_{j_k} \rightarrow U(\{j_1, j_2, \cdots, j_s\}) = U(K)$, $|K| = s$

e.g. $\begin{pmatrix}0\\1\\0\\1\end{pmatrix} \cup \begin{pmatrix}0\\0\\1\\1\end{pmatrix} = \begin{pmatrix}0\\1\\1\\1\end{pmatrix}$.

- The intersection of s' columns $C_{j_1}, C_{j_2}, \cdots, C_{j_{s'}}$ is $\bigcap_{k=1}^{s'} C_{j_k}$.

e.g. $\begin{pmatrix}0\\1\\0\\1\end{pmatrix} \cap \begin{pmatrix}0\\0\\1\\1\end{pmatrix} = \begin{pmatrix}0\\0\\0\\1\end{pmatrix}$.

Definition (d-separable).

A matrix is d-separable if for any two distinct sets of d columns D and D', $(\hat{0},\hat{1})$- $U(D) \neq U(D')$.

Definition ($\bar{d}$ - separable)

A matrix is $\bar{d}$-separable if for any two distinct sets of at most d columns D and D', i.e., $|D| \leq d$, $|D'| \leq d$, $U(D) \neq U(D')$

Definition (Disjunct matrices)

M is called d-disjunct if no column is contained in the union of any other d columns.

(•) M is d-separating if M is either a d-separable, $\bar{d}$-separable or d-disjunct matrix.

⑦′ → **Lemma** $\bar{d}$-separable ⇒ d-disjunct ⇒ $\bar{d}$-separable ⇒ d-separable. (easy)

Proof. If M is not $\bar{d}$-separable, then there D and D′ with $1 \le |D| \le d$, $1 \le |D'| \le d$, and $U(D) = U(D')$. This implies that there exists a column from D which is contained in $U(D')$ and there $U(D'')$ where $D' \subseteq D''$ and $|D''| = d$. Hence M is not d-disjunct. ∎

(Note: 頁待補.亦可。)

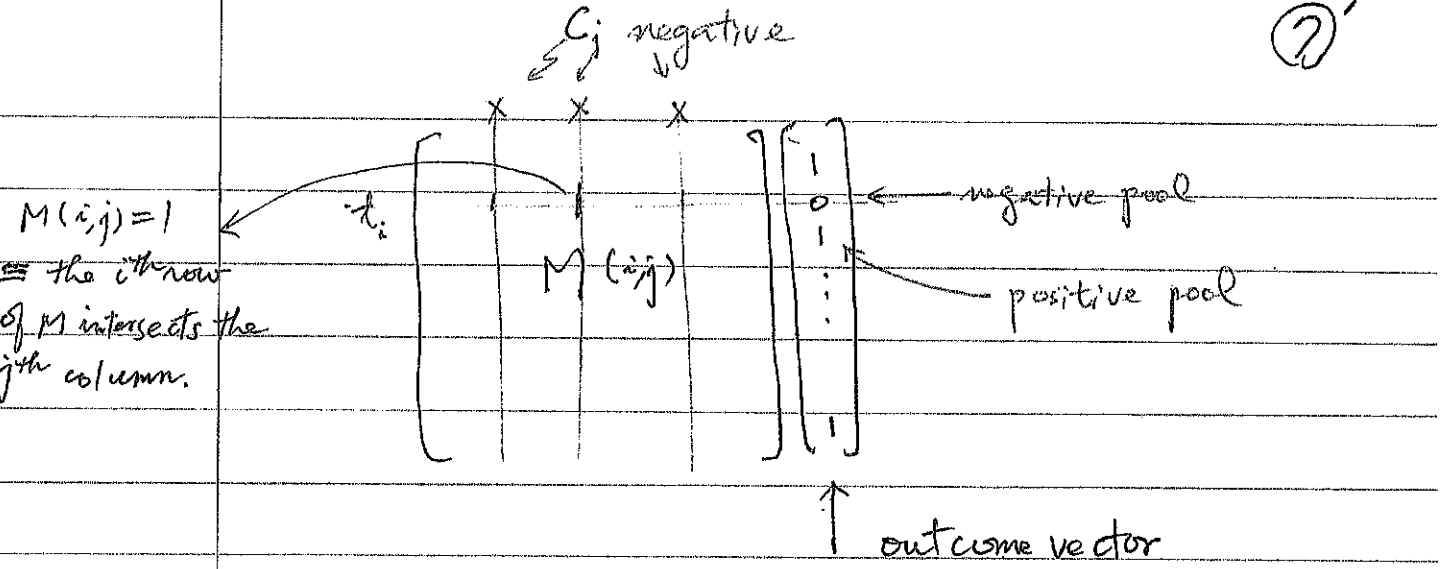
Proposition A d-separable matrix can be applied to find "exactly d positives". *Proof.* (See next page.)

Proposition $\bar{d}$-separable ⇒ to find at most d positives

→ (*) To locate the set of positives we need to check all the outcome vectors which is quite complicate. (Decoding Algorithm) is needed.

Here is an observation which is important!

↓ next page



- A test is (positive) negative if the corresponding component in the outcome vector is "0". (1)
- negative (positive) test is also called negative (positive) pool.

觀察：Negative pool 中包含的 clones 全部是 "Negative"!

結果：先利用 "negative" pools 把一些 clones 消去，再從剩下的 clones 中找 d 個即可。

(*) Hitting Set Problem

Given a set X and a collection C of subsets of X, find a minimum-cardinality subset Y of X such that $Y \cap C \neq \phi \ \forall C \in \mathcal{C}$.

Let M_D be a $t_D \times n_D$ matrix obtained from M by keeping only the set T_D of the positive pools and the set N_D of n_D clones not appearing in any negative pool.

Outcome Vector: Can be recognized as a set, V.

Denote by $t_0^V(C) = |C \setminus V|$ where C is a column.

e.g.

positives

	1	2	3	4	5	6		V
t_1	1	1	1	0	0	0		1
t_2	1	0	0	1	1	0		0
t_3	0	1	0	1	0	1		1
t_4	0	0	1	0	1	1		1
	C_1	C_2	C_3	C_4	C_5	C_6		

(•) Outcome vector plays the most important role in decoding.

$V = C_2 \cup C_3$

Note that $t_0^V(C) = |C \setminus V|$, $t_1^V = |C \cap V|$.

$t_0^V(C_1) = 1$ $\quad t_1^V(C_1) = 1$

$t_0^V(C_2) = 0$ $\quad t_1^V(C_2) = 2$

$t_0^V(C_3) = 0$ $\quad t_1^V(C_3) = 2$

$t_0^V(C_4) = 1$ $\quad t_1^V(C_4) = 1$

$t_0^V(C_5) = 1$ $\quad t_1^V(C_5) = 1$

$t_0^V(C_6) = 0$ $\quad t_1^V(C_6) = 2$

0-pool 包含了這些 items!

↑

Negatives

C_1, C_4, C_5

可存在於 ① ③ / ② ⑥ / ③ ⑥

如果是 2-disjunct 就不會是 C_2, C_3, C_6 等作判斷。

Date / 補充說明 No. ⑦''

$V = \{1, 3, 4\}$, (outcome vector) ← positives

$\mathbb{C}_p = \{\{1,3\}, \{1,4\}\}$ (positives)

$\mathbb{C} = \{$all columns$\}$

$C_1\ (C_2)\ (C_3)\ C_4\ C_5\ C_6$

$$M: \begin{matrix} t_1 \\ t_2 \\ t_3 \\ t_4 \end{matrix} \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Fact 1. $V = U(\mathbb{C}_p)$. $\quad \| \; t_o^V(C_i)$

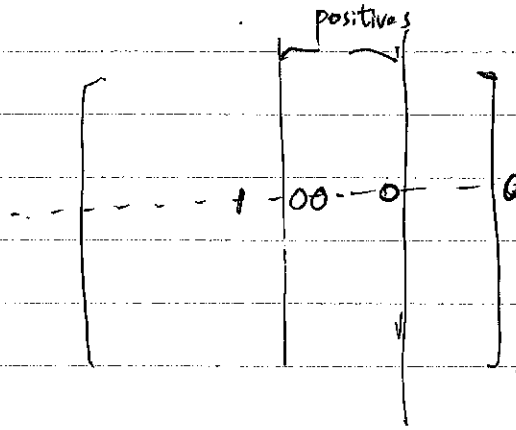
Fact 2 If a column C_i satisfying $|C_i \setminus V| > 0$, then C_i corresponds to a negative item.

Proof. All positive item corresponds to a subset of V. ∎

Fact 3 If C_i is positive, then $|C_i \setminus V| = 0$.

Fact 4. If M is d-disjunct, then there exists a linear time decoding algorithm to find the set of d positives.

Proof. By evaluating $t_o^V(C)$ for each columns in M, we conclude the proof by using Fact 2 and Fact 3.



$C_{d+1} \nsubseteq \bigcup_{j=1}^{d} C_j$

$\Rightarrow \exists$ a row i, s.t. $M(i, d+1) = 1$, but $M(i,j) = 0$, $j \in \{1, 2, \cdots, d\}$.

(•) We can use a d-disjunct matrix to identify up-to-d positive clones.

Theorem If a set of n clones which contains at most d positi
then a d-disjunct matrix can be applied to identify the positives
Moreover, the decoding algorithm is linear complexity.
of

Proof. The proof follows by evaluating $t_0^V(C)$ for each clone C. (V is the outcome vector.) We remark here that
$t_0^V(C)=0$ if C is positive and $t_0^V(C)\geq 1$ if C is negative.
($t_0^V(C)=|C\setminus V|$.)

高下一个 2-disjunct matrix

(證明) $V=\bigcup_{C \text{ is positive}} C$, 如果有一 C' 是 negative, 而 $|C'\setminus V|=0$,
則 $C'\subseteq V$, 違背 d-disjunct 的假設不符。

Algorithm (Clone Selection (N, V, D, e))

```
1   for each clone C ∈ N, ( D ← ∅ )
2       compute t_0^V(C)
3       if t_0^V(C) ≤ e
4           then D ← D ∪ {C}
5   return
```

(*) 上面定理用 Clone Selection (N, V, D, 0)。

Date / /

No. ⑦*

$d+1$ – separable $\Rightarrow$ d-disjunct

Proof. Suppose that M is not d-disjunct. Then, there exist $d+1$ columns $C_1, C_2, \cdots, C_{d+1}$, such that $C_{d+1} \subseteq \bigcup_{i=1}^{d} C_i$. This implies that $\bigcup_{i=1}^{d} C_i = C_{d+1} \cup (\bigcup_{i=1}^{d} C_i)$ and thus M is not $d+1$ – separable. ■

(•) In general, construct a d-disjunct matrix with a good ratio of d and n is very difficult.

(••) For each $d << n$, we are able to construct a d-disjunct matrix by using combinatorial design or by probabilistic method.

(*) The main tool using in probabilistic method is known as the Lovász Local Lemma.

(**) This lemma is very useful in proving the existence of certain events probabilistically.

在 ⑦*** 中大致介紹這個引理，希望它能在適當的時机被拿來應用.

Definition (Mutual independence)

An event A is said to be mutually independent of a set of events $\{B_i\}$ if for any subset β of events or their complements contained in $\{B_i\}$, we have $\Pr[A|\beta] = \Pr[A]$.

↑ conditional probability

Definition (Dependency digraph) → If the cases are of symmetric form, then a "graph".

Let $A_1, A_2, \cdots, A_n$ be events in a probability space. A directed graph $D = (V, A)$ with vertex set $V = \{1, 2, \cdots, n\}$ is a dependency digraph for $A_1, A_2, \cdots, A_n$ if for each i, A_i is mutually independent of A_j, then $(i,j) \notin A$.

(•) For symmetric case, if A_i is mutually independent of A_j, then A_j is also independent of A_i.

(•) The maximum "out-degree" (or degree in a graph) will denoted by d.

Date / /

No. 1 ***

Lovász Local Lemma (Symmetric Form)

Let $A_1, A_2, \cdots, A_n$ be a set of events with $\Pr[A_i] \le p$ for all $1 \le i \le n$ and $0 \le p < 1$. If the dependency graph has maximum degree d and $ep(d+1) \le 1$, then $\Pr\left[\bigcap_{i=1}^{n} \overline{A}_i\right] > 0$.

($e = 2.71828\cdots$, and $\overline{A}_i$ denotes the complement of A_i.)

Proof. See attached from Wiki.

(•) Lovász Local Lemma can be carried out by using an algorithm : Algorithmic L.L.L.

(•) Since it is a random type of arguments, we need the aid of random variables.

(∵) Here, we provide an explanation in next page.

(*) 主要的概念是不斷地選擇適當的 "evaluation" 來排除掉所有的 "Bad events", 如果完成, 就 "output" 它的結果, 否則如果 "p" 的選擇。