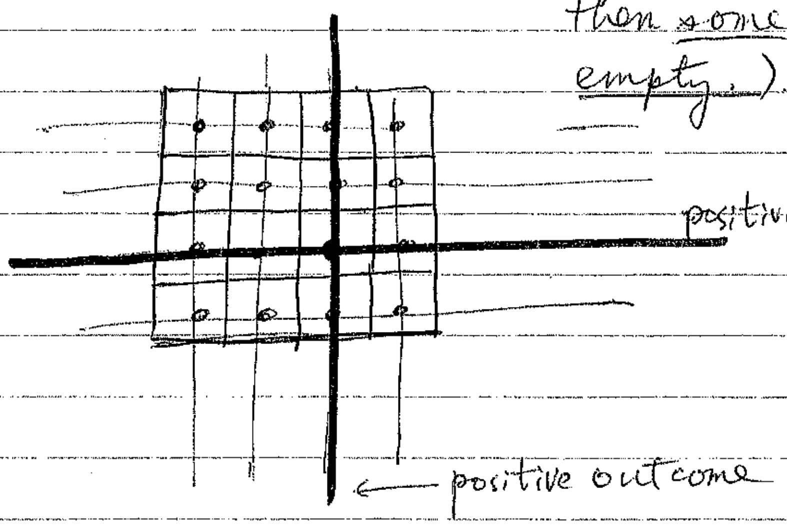
Grid Designs

For n items, choose an $m \times m$ grid where $m \geq \sqrt{n}$. Put all n items at grid points. Take each grid line as a test. If $d = 1$, then the positive item can be identified at the intersection of two lines receiving positive outcome. (Use $2m$ tests at most.)

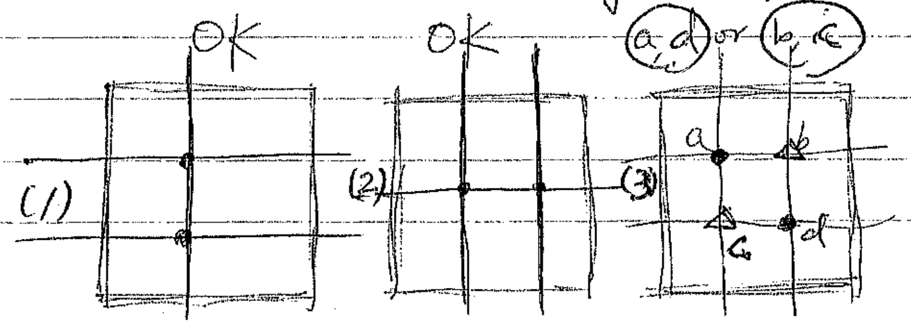
For example, $n = 16$. (If n is not a square, then some cells are empty.)



8 tests of a grid design

As a matter of fact, 8 tests are too many in the sense of optimality. But, we can extend the above design to find "two positives."

Three outcomes are possible: (1), (2) and (3)

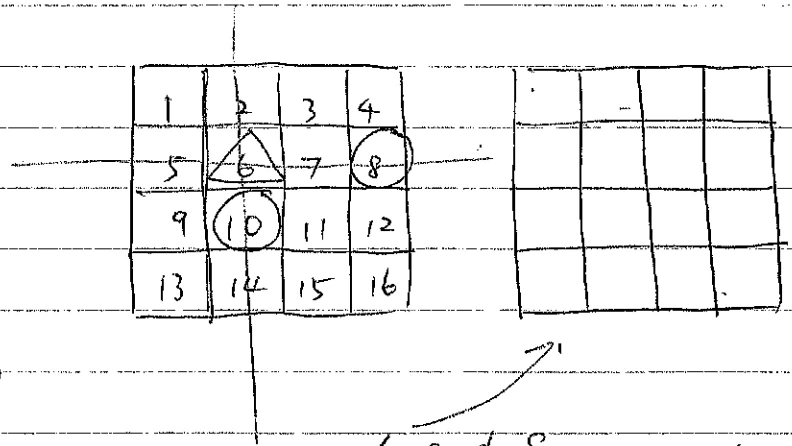


We can apply one more individual test to either a or b to decide which pair is positive for the case (3). Or, we use two grids to solve the case $d = 2$.

(C1) For any three items a, b and c, it occurs at most once that a is collinear with b on one line and also collinear with c on another line. (同行,同列只出现在一个 grid.)

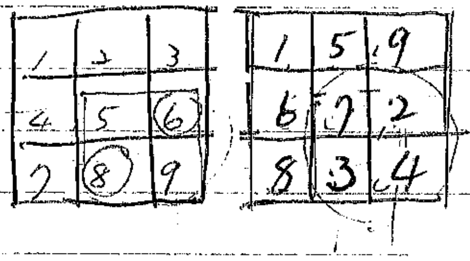
e.g. $n = 16$, $d = 2$, 8 and 10 are positives.

(*) If 8 and 10 are collinear, then they can be found easily.



6 and 8 } are not on a line
6 and 10 } simultaneously

Note that the condition (C1) is not only for 6+8, 6+10, any three (items) should satisfy the condition. So, it is left to consider how to construct such two grids. The set of grids we construct is known as a <u>grid-design</u>.

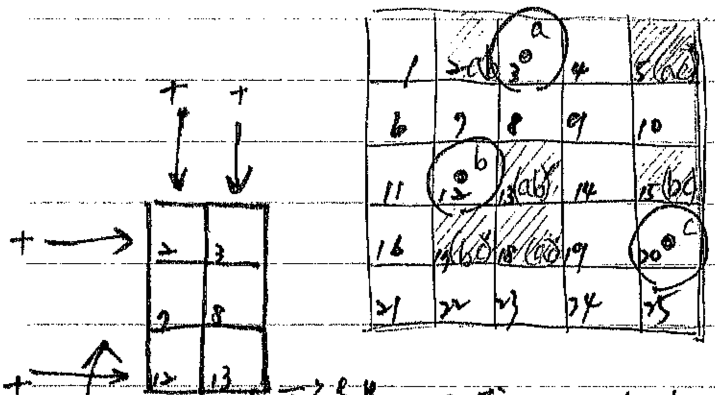


An example, $n=9$, $d=2$.

Since any two items are collinear in one of the two grids, we can find the two positives easily. (?)

Now, consider $d>2$. For example, $d=3$.

假设 3, 12, 20 are positives.



可能是 2,13 或 3,12. (check 2 或 13 即可)

(*) We need an extra grid (satisfying (C1)) to identify the items $\boxed{ab \text{ and } (ab)'}$ ac and (ac)', bc and (bc)' respectively. (?)
同理

(**) $\binom{d}{2}+1$ grids are needed if only (C1) condition is needed for the grid. (?)

(∘) The intersection of 4 suspect sets ($d=3$) is the set of positives.

(∘∘) In case that for $d=3$, we only obtain (<6) outcomes which are positive, then the number of grids can be smaller.

But, in general, we have no idea where are the positives.

(*) We use $\left[\binom{d}{2}+1\right]\cdot\lceil\sqrt{n}\rceil\cdot 2$ tests in grid design (C1 condition)!

<u>A better idea of design</u>

(C2) Every two items can be collinear in <u>at most</u> one grid.
(Unique Collinearity Condition)

<u>Theorem</u> (F.K. Hwang, JGT(19), 1995, no. 3, 333–337).

$\left\lceil \frac{d+1}{2} \right\rceil$ grids satisfying (C2) are enough to identify d positives.

It is worth of noting here that we can use these grids to construct a d-disjunct matrix. This is why this grid design is beautiful.

(*)

<u>Proof.</u> Suppose not. Then, there exists a negative item which can not be identified by $\left\lceil \frac{d+1}{2} \right\rceil$ grids. By the explanation earlier, this item must be collinear with at least two positives in each grid. By (C2), these positives are distinct. Hence, in total, we have at least $\left\lceil \frac{d+1}{2} \right\rceil \cdot 2$ positives, a contradiction. ∎

(d).

Example $d = 2$ (page 3)

1	2	3		1	5	9
4	5	6		6	7	2
7	8	9		8	3	4

An example satisfying (C 2).

By Theorem (Kwang), two such grids are enough for identifying "two" positives.
(three)

(◦) For example, if 5 is negative and we are not able to identify "5" since 5 is collinear with positives "6" and "8" as mentioned earlier ~~##~~. But, in the second grid, 5 is not collinear with "6", and "8", by (C 2), hence 5 can be identified as "negative". (If "1" is also positive, we can determine "1" as well.)

Another fact

If we take each row and each column of the above two grids as a set, then we have an affine plane of order 3.

$$\mathcal{B} = \left\{ \begin{array}{l} \{1,2,3\}, \{4,5,6\}, \{7,8,9\}, \{1,4,7\}, \{2,5,8\}, \{3,6,9\} \\ \{1,5,9\}, \{2,6,7\}, \{3,4,8\}, \{1,6,8\}, \{3,5,7\}, \{2,4,9\} \end{array} \right\}, \quad X = \{1, 2, \cdots, 9\}.$$

(•) $r \times c$ grid: A grid with r rows and c columns.

(C3) Every two items are collinear in exactly one grid.

Definition (Grid-Block Designs)

An $r \times c$ grid-block design of order n is a pair $(X, \mathbb{B})$ where $|X| = n$ and $\mathbb{B}$ is a set of $r \times c$ grids such that every two elements of X are collinear in exactly one grids in $\mathbb{B}$.

(•) The example mentioned in 5 is a 3×3 grid-block design of order 9.

(*) If an $r \times c$ grid-block design of order n exists, then

(1) $n \geq r \cdot c$,

(2) $r + c - 2 \mid n - 1$, and (N.C.)

(3) $r \cdot \binom{c}{2} + c \cdot \binom{r}{2} \,\Big|\, \binom{n}{2}$.

(•) In fact, the size of $\mathbb{B}$, $|\mathbb{B}| = \dfrac{\binom{n}{2}}{r \cdot \binom{c}{2} + c\binom{r}{2}}$.

Problem The above N.C. is also sufficient in constructing a grid-block design.

Construction of grids satisfying (C2) when $n = k^2$

0	0	0
1	1	1
2	2	2

0	1	2
0	1	2
0	1	2

0	1	2
2	0	1
1	2	0

0	2	1
2	1	0
1	0	2

orthogonal 3x3 arrays

$\Downarrow$

00	01	02
10	11	12
20	21	22

00	12	21
22	01	10
11	20	02

orthogonal Latin Squares
(How to construct?)
Next Semester

$(i,j) \longrightarrow 3i+j$

Grids

0	1	2
3	4	5
6	7	8

0	5	7
8	1	3
4	6	2

One way to construct. But, in general, the idea mentioned in 6'' is much better! (later)

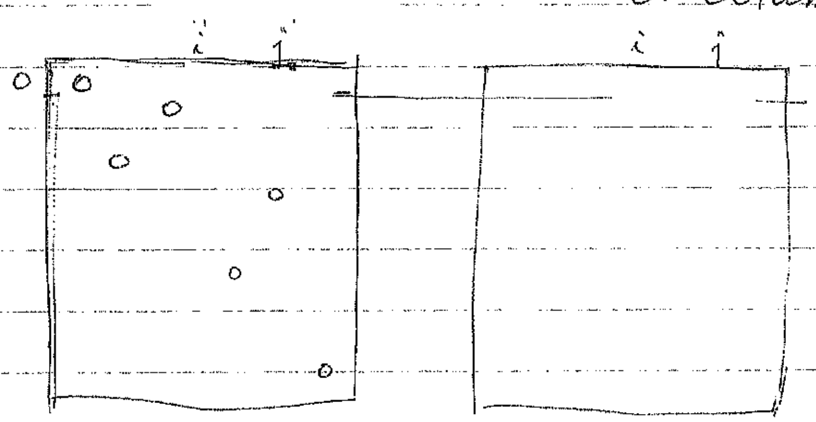
Theorem (Hwang)

There exist g $k \times k$ grids satisfying (C2) if and only if there exist $2(g-1)$ mutually orthogonal Latin squares of order k ($2g$ orthogonal arrays of order k).

Proof. See 6'.

Let the set of $2(g-1)$ MOLS(k) be $L_1, L_2, \dots, L_{2(g-1)}$ and the two index-arrays be L_r and L_c. Now, by combining two arrays L_r and L_c, L_1 and L_2, $\dots$, L_{2g-3} and L_{2g-2} (See 6'' for refrence) we obtain g grids, $G_0, G_1, \dots, G_{g-1}$. Now, it suffices to show that these grids do satisfy (C2).

By orthogonality, each element of $\mathbb{Z}_{k^2}$ $\{0, 1, \dots, k^2-1\}$ occurs exactly once in each grid. Now, consider two elements x and y in $\mathbb{Z}_{k^2}$. Let x and y be collinear in G_ℓ without loss of generality. Say, in the first row. Then, they have the same entry "0" in L_1. Now, we claim that x and y can not be collinear in the other grids. Both cases for x and y in the same row or column will lead to a contradiction.



L_1 L_2.

See the idea in 6'' and 6'''.

(*) The idea of combining two Latin squares into one grid
(orthogonal)

e.g.

	0	1	2	3	4
0	0	1	2	3	4
1	4	0	1	2	3
2	3	4	0	1	2
3	2	3	4	0	1
4	1	2	3	4	0

$\perp$

	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

⇓

	0	1	2	3	4
0	00	33	11	44	22
1	23	01	34	12	40
2	41	24	02	30	13
3	14	42	20	03	31
4	32	10	43	21	04

← 左边填0,右边分别填0,1,2,3,4的位置

← 左边填2, ...

上面⟨2,3⟩出现在(3,0)的位置

↑ 右边填1

(·) 由於垂直,填入grid的item都不相同,而且每一行,每一列填入的"Entries",左,右都维持不同的元素。

(··) 這个Grid的特性是,同一列的Item來自左边拉丁方陣相同的元素,例如第三列,左边填"2"。來自同一行的右边皆填"1"。
(第二行)

(···) 証明的概念是如下:

$$\underbrace{\boxed{L_1} \perp \boxed{L_2}}_{grid(1)} \qquad \underbrace{\boxed{L_3} \perp \boxed{L_4}}_{grid(2)}$$

↓ 6'''

Proof.

If two items are collinear in grid(1) and grid(2), then we have (1) row and row, $\Rightarrow L_1 \not\subset L_3$; and

(2) row and column $\Rightarrow L_2 \not\subset L_4$. $\rightarrow\leftarrow$

■

(*) Again, we would like to mention one more time that the grid design does provide a construction of d-disjunct matrices. For larger d, the construction is getting closer to the informatic lower bound.

(**) Grid designs can be applied to Group Testings with bounded pool size which is very important in pratical applications.

Theorem (Well-known?)

If k is a prime power, there exist $k-1$ mutually orthogonal Latin squares of order k. (Use finite field $GF(k)$!)

For example, if $k=3$, there exist two orthogonal Latin squares of order 3. And if $k=9$, there are "8" MOLS(9).

(•) $n = k^2$, $k=9$, $2(g-1)=8$, $g=5$.

$\lceil \frac{d+1}{2} \rceil$ grids are enough for identifying d positives.

$\Rightarrow$ "$d \le 9$" positives in 81 items can be identified by using at most $5 \cdot 2k = 90$ tests. ($2k$ tests for each array!)

(••) Arrange each line (test) as a row, we obtain a d-disjunct matrix. (Bounded row size!!)

Remark In general, $k^2 \ll n$ or $r \times c \ll n$.
(r rows and c columns)

In this situation, we have an $r \times c$ grid design. Again, it's a design problem or equivalently a graph decomposition problem. (Or a graph packing problem!)