Lecture 9
Topic 7: Incomplete Block Designs: Latin Squares (LS)

Blocking

...can account for heterogeneity among EU's in our system
...thereby allowing us to compare treatments under more uniform conditions
...and reducing experimental error, increasing precision

If one blocking variable is good, is two even better?

Imagine a tasting panel with four judges, evaluating four high-quality Granite State ciders:

The judges	The ciders
Tom	Apple Hill
Jane	Blossom Orchards
Clyde	Red Thunder
Monique	Treeshine

Where are some likely sources of noise in a panel like this?

If each judge tastes all 4 ciders: 1 day, 16 total tastings
If each judge tastes all 4 ciders, in all possible ranks: 4 days, 64 total tastings

The problem with including another class variable (in this case, another blocking variable) is that the size of the experiment balloons, unless we relax the requirement of *completeness*.

Incomplete Block Designs

EU's are grouped into "blocks" that are not large enough to contain all combinations of all other factors.

Advantage

Helps address heterogeneity

Disadvantage

Randomization, layout, and analysis get more complicated

There are many different kinds of incomplete block designs, distinguished by:

	Latin Square	α-Lattice
Number of blocking criteria	2	1
Balanced or partially balanced	Balanced	Partially
Resolvability	N/A	Resolvable
Shape (square, rectangular, etc.)	Square	No constraint
Process for generating the design	Cyclic	Cyclic

Balance

In a balanced incomplete block design, *each treatment occurs in the same block with every other treatment an equal number of times* (λ, associate class).

Block	Rep I	Rep II	Rep III	Rep IV
1	1 2 3	1 4 7	1 5 9	1 6 8
2	4 5 6	2 5 8	2 6 7	2 4 9
3	7 8 9	3 6 9	3 4 8	3 5 7

In this case (a basic square lattice), λ = 1.

The problem: To achieve balance, $r = \lambda \frac{t-1}{k-1}$

t = the number of treatments (genotypes)
k = the number of experimental units in each block (block size)
b = total number of blocks in the experiment
r = number of replicates of each treatment
λ = number of times that treatments occur together in the same block
N = total number of experimental units

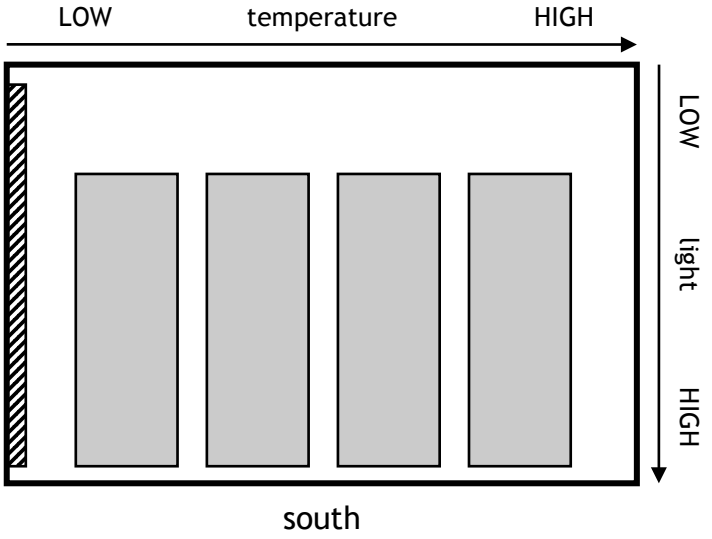
The constraints are that k, b, r, and λ must be integers, and N = b * k = r * t

Example: For t = 10 genotypes, with a block size of k = 4, we obtain:

r = 6, b = 15, λ = 2
N = 6 * 10 = 60

For a Latin Square, r = λ = t = k; so N = t^2.

Latin Squares (LS) are experimental designs used to remove variation from the error through the use of *two* separate blocking variables.

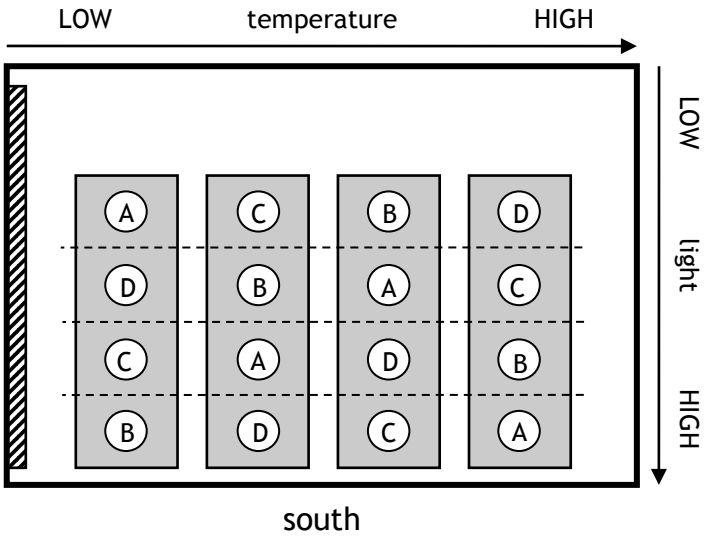


The two blocking variables are referred to as rows and columns.

These are highly constrained designs: # rows = # columns = # treatments = t

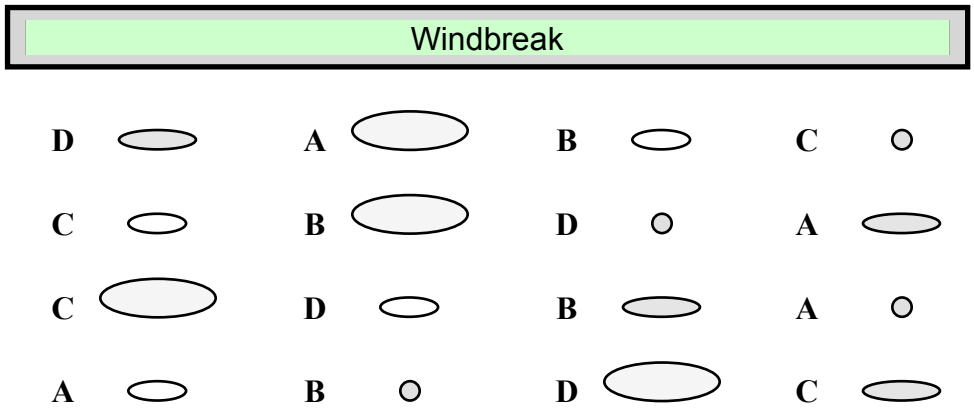
There is one replication (EU) per row-column combination.

Treatments are assigned to EU's such that each treatment level appears exactly once in each row and exactly once in each column.



Examples

1. A 4 x 4 Latin Square with treatments (A, B, C, D) randomized according to two blocking variables: Tree size and distance from a windbreak.



Rows and columns do not necessarily refer to the spatial distribution of the experimental units. They can also refer to the order in which treatments are performed, to different pieces of equipment used in the experiment, to different technicians taking the measurements, etc.

2. A 3 x 3 Latin Square with treatments (Wines A, B, C) randomized according to two blocking variables: Judge and day of testing.

Day	**Judge**		
	Joe	**Laura**	**Rose**
Monday	B	C	A
Wednesday	C	A	B
Friday	A	B	C

3. A 3 x 3 Latin Square with treatments (Wines A, B, C) randomized according to two blocking variables: Judge and *sequence* of tasting.

Sequence	**Judge**		
	Joe	**Laura**	**Rose**
First	B	C	A
Second	C	A	B
Third	A	B	C

Randomization (manual)

1. Choose a **standard square** of the appropriate dimensions.

2x2

A	**B**
B	A

3x3

A	**B**	**C**
B	C	A
C	A	B

4x4

A	**B**	**C**	**D**
B	A	D	C
C	D	B	A
D	C	A	B

A	**B**	**C**	**D**
B	C	D	A
C	D	A	B
D	A	B	C

A	**B**	**C**	**D**
B	D	A	C
C	A	D	B
D	C	B	A

A	**B**	**C**	**D**
B	A	D	C
C	D	A	B
D	C	B	A

As t increases, the number of possible standard squares increases rapidly.

For any given square size (t x t), the number of possible different Latin squares is:

(# of standard squares) (t!) (t - 1)!

For example, for t = 4, the number of total possible unique squares is: (4) (4!) (3!) = 576.

2. Select two sets of random numbers with size equal to the treatments involved. Then assign ranks to these sets of random numbers. For example:

	Set 1 (for columns)	**Set 2 (for rows)**
Random number	9 1 6 4	8 5 3 7
Rank	4 1 3 2	4 2 1 3

3. a. Assign the ranks to the rows and column headers of the standard square chosen in Step 1.
b. Order the rows according to rank.
c. Order the columns according to rank.

a. Assign ranks

	4	**1**	**3**	**2**
4	A	B	C	D
2	B	C	D	A
1	C	D	A	B
3	D	A	B	C

b. Order rows

	4	**1**	**3**	**2**
1	C	D	A	B
2	B	C	D	A
3	D	A	B	C
4	A	B	C	D

c. Order columns

	1	**2**	**3**	**4**
1	D	B	A	C
2	C	A	D	B
3	A	C	B	D
4	B	D	C	A

4. Randomly assign the generic treatment ID's (A – D) to the 4 treatments.

Randomization (using R technology)

```
#This script defines a function Latin() that can be used to generate LS randomizations

Latin <- function (x)
{
  LS = matrix(LETTERS[1:x], x, x)
  LS = t(LS)
  for (i in 2:x) LS[i, ] = LS[i, c(i:x, 1:(i - 1))]
  for (i in 1:20) {
      LS = LS[sample(x), ]
      LS = LS[, sample(x)]
  }
  LS
}

#Example: For a 5x5 LS:
Latin(5)
```

Output:

```
> Latin(5)
     [,1] [,2] [,3] [,4] [,5]
[1,] "B"  "E"  "A"  "C"  "D"
[2,] "A"  "D"  "E"  "B"  "C"
[3,] "C"  "A"  "B"  "D"  "E"
[4,] "E"  "C"  "D"  "A"  "B"
[5,] "D"  "B"  "C"  "E"  "A"
```

The linear model

The linear model for the Latin Square:

$$Y_{(i)jk} = \mu + \tau_{(i)} + \rho_j + \kappa_k + \varepsilon_{(i)jk}$$

The parentheses around the index "i" is due to the incomplete nature of this design; in any given row-column combination, only one treatment level occurs. In dot notation:

$$\sum_{i,j}^{r}(\bar{Y}_{ij} - \bar{Y}_{..})^2 = r\sum_{i=1}^{r}(\bar{Y}_{i} - \bar{Y}_{..})^2 + r\sum_{j=1}^{r}(\bar{Y}_{j.} - \bar{Y}_{..})^2 + r\sum_{k=1}^{r}(\bar{Y}_{.k} - \bar{Y}_{..})^2 + \sum_{i,j}^{t}(Y_{ij} - \bar{Y}_{i} - \bar{Y}_{j.} - \bar{Y}_{.k} + 2\bar{Y}_{..})^2$$

$$\text{TSS} \quad = \quad \text{SST} \quad + \quad \text{SSR} \quad + \quad \text{SSC} \quad + \quad \text{SSE}$$

The generic ANOVA table for a Latin Square looks like this:

Source	df	SS	MS	F
Rows	r - 1	SSR	SSR/(r-1)	MSR/MSE
Columns	r - 1	SSC	SSC/(r-1)	MSC/MSE
Treatments	r - 1	SST	SST/(r-1)	MST/MSE
Error	(r-1)(r-2)	SSE	SSE/(r-1)(r-2)	
Total	r^2-1	TSS		

Example: A 4x4 LS designed to compare the yield of 4 wheat varieties (A – D). [ST&D 230]

	Column			
Row	**1**	**2**	**3**	**4**
1	C: 10.5	D: 7.7	B: 12.0	A: 13.2
2	B: 11.1	A: 12.0	C: 10.3	D: 7.5
3	D: 5.8	C: 12.2	A: 11.2	B: 13.7
4	A: 11.6	B: 12.3	D: 5.9	C: 10.2

Source	DF	SS	MS	F Value	Pr > F
ROW	3	1.955	0.652	1.44	**0.322 NS**
COL	3	6.800	2.267	5.00	**0.045 ***
TRTMT	3	78.925	26.308	58.03	**0.000 *****
Error	6	2.720	0.453		
Total	15	90.400			

4 sources of variation: 3 due to design, 1 due to error

```
#Example, unreplicated Latin Square

#read in, re-classigy, and inspect the data
LS_dat<-as.data.frame(LS_dat)
LS_dat$Row<-as.factor(LS_dat$Row)
LS_dat$Col<-as.factor(LS_dat$Col)
str(LS_dat, give.attr = F)

#The ANOVA
LS_mod<-lm(Response ~ Row + Col + Trtmt, LS_dat)
anova(LS_mod)

#TESTING ASSUMPTIONS
#Generate residual and predicted values
LS_dat$resids <- residuals(LS_mod)
LS_dat$preds <- predict(LS_mod)

#Look at a plot of residual vs. predicted values
plot(resids ~ preds, data = LS_dat,
     xlab = "Predicted Values",
     ylab = "Residuals")

#Perform a Shapiro-Wilk test for normality of residuals
shapiro.test(LS_dat$resids)

#Perform Levene's Test for homogenity of variances among treatment levels
#library(car)
leveneTest(Response ~ Trtmt, data = LS_dat)

#3 separate Tukey nonadditivity tests are required for the 3 possible
#interactions R*C, R*T, C*T.
#What follows is the test for the Row*Treatment interaction:

#Testing for significance of the Row*Trtmt interaction
LS_RT_mod<-lm(Response ~ Row + Trtmt, LS_dat)
RT_preds <- predict(LS_RT_mod)
LS_dat$RT_sqpreds <- RT_preds^2

LS_RT_Tukey_mod<-lm(Response ~ Row + Trtmt + RT_sqpreds, LS_dat)
anova(LS_RT_Tukey_mod)
```

The Latin Square design removes from the MSE variation originating from two separate sources (blocking variables). This can increase test sensitivity.

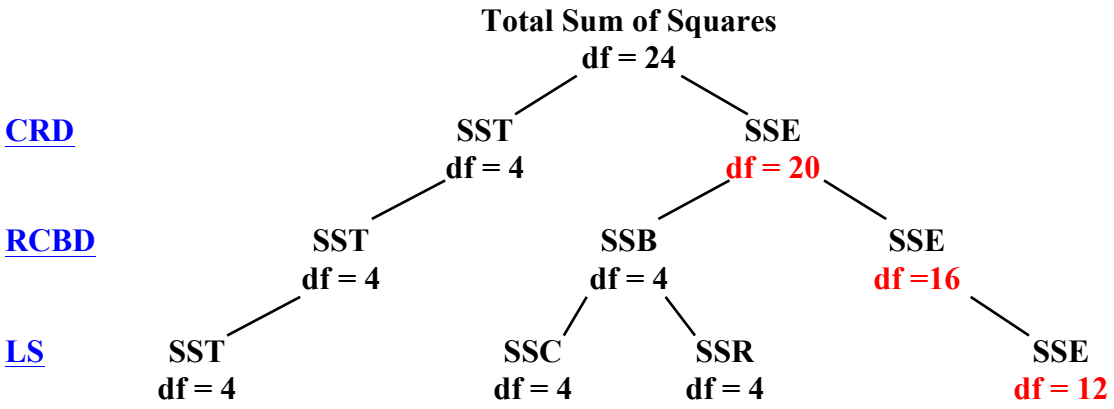
Disadvantages:

1. The severe design constraint (t = r = c) is inconvenient for experimental work, particularly when the number of treatments is large.

2. The design is only valid if there are no interactions among rows, columns, and treatments.

3. Squares smaller than 4 x 4 generally have too few replications for a desirable level of precision (i.e. too few error degrees of freedom).

Assume 5 treatments are assigned to 25 experimental units, with equal replication per treatment.



This diagram illustrates the loss of degrees of freedom from the experimental error as the chosen design becomes more complex.

Unless there is enough variation in the experimental error of the CRD that can be allocated to the chosen blocking variable(s) to compensate for the loss of error degrees of freedom, the resulting design will be less efficient.

A numerical example of the effect of the missing df_e on the critical F value:

3x3 Latin Square	**5x5 Latin Square**
LS $df_e = 2$ (RCBD $df_e = 4$)	LS $df_e = 12$ (RCBD $df_e = 16$)
LS $F_{0.05,2,2} = 19.0$	LS $F_{0.05,2,12} = 3.89$
RCBD $F_{0.05,2,4} = 6.94$	RCBD $F_{0.05,2,16} = 3.63$

This loss of df_e has a huge effect on the sensitivity of small squares!

Relative Efficiency

All numbers taken from the wheat example on page 6: $MSE_{LS} = 0.45$, $MSR_{LS} = 0.65$, and $MSC_{LS} = 2.27$.

If **columns** were removed as a blocking variable:

$$\hat{MSE}_{RCBD} \cong \frac{df_{col} * MSC_{LS} + (df_{trt} + df_{error})MSE_{LS}}{df_{col} + df_{trt} + df_{error}} = \frac{3*2.27 + (3+6)\ 0.45}{3+3+6} = 0.91$$

$$RE_{LS:RCBD} = \frac{(df_{e(LS)} + 1)(df_{e(RCBD)} + 3)MSE_{RCBD}}{(df_{e(RCBD)} + 1)(df_{e(LS)} + 3)MSE_{Ls}} = \frac{(6+1)(9+3)0.91}{(9+1)(6+3)0.45} = 1.89$$

The column grouping increased the precision of the model by an estimated 89%. This parallels the significant effect of columns identified in the F tests.

If **rows** were removed as a blocking variable:

$$\hat{MSE}_{RCBD} \cong \frac{df_{row} * MSR_{LS} + (df_{trt} + df_{error})MSE_{LS}}{df_{row} + df_{trt} + df_{error}} = \frac{3*0.65 + (3+6)\ 0.45}{3+3+6} = 0.50$$

$$RE_{LS:RCBD} = \frac{(6+1)(9+3)0.50}{(9+1)(6+3)0.45} = 1.04$$

The row blocking increased the precision of the model only by an estimated 4%. This parallels the non-significant effect of rows identified in the F tests.

Repeated Latin squares

To increase the degrees of freedom for experimental error, Latin Squares may be replicated. This strategy is known as repeated Latin Squares, and each individual Latin Square can be thought of as an independent replication of the entire experiment.

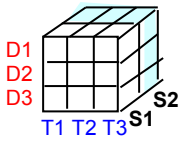
There are four scenarios, depending on how the replicated squares relate to one another:

1. They share the same rows but not the same columns (commons rows, independent cols)
2. They share the same columns but not the same rows (commons cols, independent rows)
3. They share the same rows and the same columns (commons rows and cols)
4. They share neither the same rows nor the same columns (independent rows and cols)

Example: Three gasoline additives (TREATMENTS: A,B,C) were tested for gas efficiency by three drivers (ROWS: 1,2,3) using three different tractors (COLUMNS: 1,2,3). The variable measured was the yield of carbon monoxide in a trap. The experiment was repeated on two separate days (SQUARES: 1,2). The possible replication scenarios:

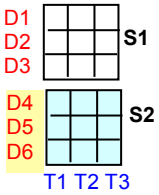
Case 1: Same tractors (col) and same drivers (row)

```
LS1_mod<-lm(CO ~ Day + Tractor + Driver + Trtmt, LS_dat)
anova(LS1_mod)
```



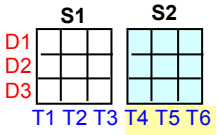
Case 2: Same tractors (col) but different drivers (row)

```
LS4_mod<-lm(CO ~ Day + Tractor + Day:Driver + Trtmt, LS_dat)
anova(LS4_mod)
```



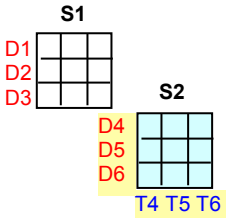
Case 3: Different tractors (col) but same drivers (row)

```
LS3_mod<-lm(CO ~ Day + Day:Tractor + Driver + Trtmt, LS_dat)
anova(LS3_mod)
```



Case 4: Different tractors (col) and different drivers (row)

```
LS4_mod<-lm(CO ~ Day + Day:Tractor + Day:Driver + Trtmt, LS_dat)
anova(LS4_mod)
```



More complicated Latin squares (Box, Hunter, & Hunter Chapter 8)

To control more than two independent sources of variability, more complicated designs like the Graeco-Latin Square (GLS) or the Hyper-Graeco-Latin Square (HGLS) may be useful.

The comparison of 4 different oil additives (Treatments A - D), blocked by 4 different drivers (I - IV), 4 different cars (1- 4), and 4 different days (α - δ).

	Car			
Driver	**1**	**2**	**3**	**4**
I	Aα	Bβ	Cγ	Dδ
II	Bδ	Aγ	Dβ	Cα
III	Cβ	Dα	Aδ	Bγ
IV	Dγ	Cδ	Bα	Aβ

Here, a quadruple-replicated 4x4 Latin Square could accomplish a similar objective (Squares = Days) but would require four times the resources (64 driving tests, 16 on each of four days, vs. 16 driving tests, 4 on each of four days).

On the other hand, replicated Latin Squares with shared cars and drivers would provide **much more power** (Replicated LS $df_e = 51$ vs. GLS $df_e = 3$).