1 Experimental Design

Experimental design

Experimental design can be thought of as the study of careful perturbation.
Many hypotheses can be tested through observation of natural phenomena, but all non-trivial inference from observation is correlative.
Through manipulation of natural phenomena, it is possible to tease causality from correlation

Objectives

The first step in experimental design is to clearly state the hypothesis and in the specific predictions from that hypothesis. It is often tempting to try a complex analysis to answer a simple question.
Example: *Completely randomized design with one factor* . This is a design where you are interested in the effect of a single factor and you can perform all the replicates of all the treatment levels at one time.

- Can test for differences among means of the treatment or factor levels.

Factors

Factors are discrete categorical variables. They can be ordered or unordered. Factors are typically used as the independent variable in Anova

```
> vec <- c(1, 2, 3, 2, 5, 5, 1, 3)
> vec

[1] 1 2 3 2 5 5 1 3

> vec2 <- as.factor(vec)
> vec2

[1] 1 2 3 2 5 5 1 3
Levels: 1 2 3 5

> is.factor(vec)

[1] FALSE

> is.factor(vec2)

[1] TRUE
```

Ordered factors

```
> vec3 <- ordered(vec)
> vec3

[1] 1 2 3 2 5 5 1 3
Levels: 1 < 2 < 3 < 5

> is.factor(vec3)

[1] TRUE
```

Basic process

- assign individuals to experimental units
- assign units to treatments (manipulations)
- arrange treatments through time and/or space
- arrange data collection through time and/or space

Assigning individuals

As mentioned before, individuals must be assigned to the experimental units at random.

Replication

For each level of a factor, usually multiple *replicates* are performed.
Multiple Replicates:

- Give better estimates of τ_i
- Allow estimation of ϵ in linear models (and comparison to expected distribution)
- Allow statistical tests

Formal expression of randomized design with one factor

$$Y_{ij} = \mu + \tau_i + \epsilon_{ij} \tag{1}$$

This should look familiar. The important parameters for a completely randomized, one factor design are:

- f, the number of factors (here f is defined as 1)
- k, the number of levels for a factor (i ranges from $1 - k$)
- n, number of replications per level of the factor (j ranges from $1 - n_i$)

The total sample size for the experiment is: $f \times k \times n$ This number is relevant in terms of experimental cost but not necessarily in terms of statistical power. If k approachs n in this design, it is possible to lose power.

Assignment of units to treatments

In a completely randomized design, both the order of treatment level applied and the assignment of experimental units to treatments are randomized.

Why randomize

The effect of randomization is to eliminate correlation that may exist among replicates or experimental units.

- changes in functionality of instrument (DNA flurometer)
- genetic relatedness of experimental subjects

The variation in the total experiment that might be due to that correlation gets subsumed into the ϵ term.
One implication of this is that both random *and* systematic errors tend to increase ϵ and reduce F.

2 Blocking designs

Complete randomized w/ blocking

Imagine a situation where you cannot perform all of your treatments at one time

- limited incubator space
- limited tank space
- limited time

If it is possible to break up the experiment into subgroups that can accomodate every treatment, the subgroups can be used as *blocks* .

Blocking cont

Blocking allows the effects of random variation over experimental runs to be explicitly modeled. This takes some of the variation to be explained by the model out of the within-treatment error term.

Complete randomized block design

$$Y_{ijl} = \mu + \text{block}_l + \tau_i + \epsilon_{ijk}$$

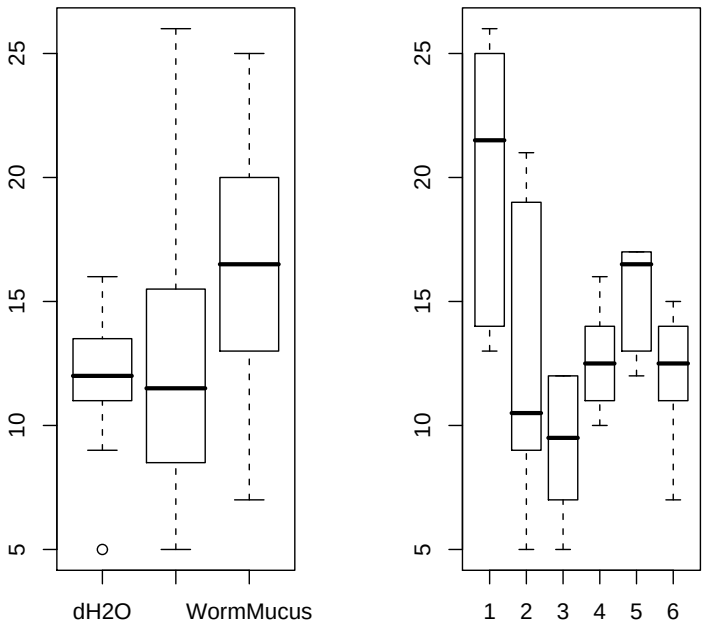
Snake data

```
> snake <- as.factor(rep(rep(1:6, 3), 2))
> flicks <- c(13, 9, 17, 10, 13, 11, 18, 19, 12, 16, 17, 14, 8,
+     12, 10, 11, 12, 12)
> flicks <- c(flicks, flicks + round(rnorm(18, 0, 4), 0))
> stimulus <- as.factor(c(rep("FishMucus", 6), rep("WormMucus",
+     6), rep("dH2O", 6), rep("FishMucus", 6), rep("WormMucus",
+     6), rep("dH2O", 6)))
> snakedat1 <- data.frame(snake, stimulus, flicks)
> snakedat2 <- snakedat1
> snakedat2[snakedat2$snake == "1", 3] <- snakedat2[snakedat2$snake ==
+     "1", 3] + 6
> snakedat2[snakedat2$snake == "3", 3] <- snakedat2[snakedat2$snake ==
+     "3", 3] - 5
> snakedat2

   snake  stimulus flicks
1      1 FishMucus     19
2      2 FishMucus      9
3      3 FishMucus     12
4      4 FishMucus     10
5      5 FishMucus     13
6      6 FishMucus     11
7      1 WormMucus     24
8      2 WormMucus     19
9      3 WormMucus      7
10     4 WormMucus     16
11     5 WormMucus     17
12     6 WormMucus     14
13     1      dH2O     14
14     2      dH2O     12
```

```
15     3      dH2O      5
16     4      dH2O     11
17     5      dH2O     12
18     6      dH2O     12
19     1 FishMucus     14
20     2 FishMucus      8
21     3 FishMucus     11
22     4 FishMucus      5
23     5 FishMucus     15
24     6 FishMucus     12
25     1 WormMucus     28
26     2 WormMucus     11
27     3 WormMucus      0
28     4 WormMucus      9
29     5 WormMucus     12
30     6 WormMucus     11
31     1      dH2O     18
32     2      dH2O     16
33     3      dH2O     -3
34     4      dH2O      9
35     5      dH2O     11
36     6      dH2O     12
```

Snake data-transmogrified

```
> par(mfrow = c(1, 2))
> boxplot(flicks ~ stimulus, data = snakedat2)
> boxplot(flicks ~ snake, data = snakedat2)
```

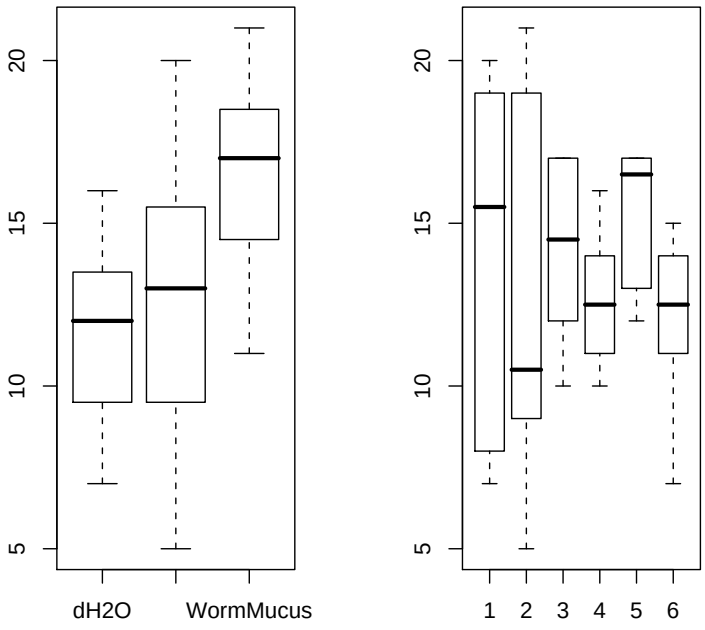


Snake data analyzed

```
> summary(aov(flicks ~ stimulus, data = snakedat2))

            Df  Sum Sq Mean Sq F value Pr(>F)
stimulus     2   68.39   34.19   1.036 0.3661
Residuals   33 1089.17   33.01

> summary(aov(flicks ~ stimulus + snake, data = snakedat2))

            Df Sum Sq Mean Sq F value    Pr(>F)
stimulus     2  68.39   34.19  2.1311    0.1376
snake        5 639.89  127.98  7.9759 8.925e-05 ***
Residuals   28 449.28   16.05
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

Snake data-actual

```
> par(mfrow = c(1, 2))
> boxplot(flicks ~ stimulus, data = snakedat1)
> boxplot(flicks ~ snake, data = snakedat1)
```



Snake data analyzed, cont

```
> summary(aov(flicks ~ stimulus + snake, data = snakedat1))

            Df Sum Sq Mean Sq F value Pr(>F)
stimulus     2  68.39   34.19  2.1311 0.1376
snake        5  66.22   13.24  0.8254 0.5423
Residuals   28 449.28   16.05
```

If no effect of block, then subsequent experiments can ignore block and just randomly assign treatment replicates.

Latin square

There are times in an experimental design where there there are clearly defined gradients built into the design. For example, there may be a field that has a gradient of moisture and

a gradient of nitrogen running perpendicular. The Latin Square design is a special type of complete blocking where 2-dimensional space is explicitly taken into account. It can also be used anytime there are two nuisance factors that should affect results.

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

Rabbit blisters

```
> rabbits <- read.csv("rabbits.csv", header = T)
> rmat <- matrix(rabbits$Order, ncol = 6)
> rownames(rmat) <- rabbits$Position[1:6]
> colnames(rmat) <- unique(rabbits$Rabbit)
> rmat

  1 2 3 4 5 6
1 3 5 4 1 6 2
2 4 2 6 5 3 1
3 1 3 5 6 2 4
4 6 1 3 2 4 5
5 2 4 1 3 5 6
6 5 6 2 4 1 3
```

Rabbit blisters cont

```
> summary(aov(Observation ~ as.factor(Rabbit) + as.factor(Position) +
+     as.factor(Order), data = rabbits))

                    Df  Sum Sq Mean Sq F value  Pr(>F)
as.factor(Rabbit)    5 12.8333  2.5667  3.9096 0.01235 *
as.factor(Position)  5  3.8333  0.7667  1.1678 0.35919
as.factor(Order)     5  0.5633  0.1127  0.1716 0.97013
Residuals           20 13.1300  0.6565
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

Incomplete randomized blocked design

There are frequently times where there are more treatments than can be placed within a block. In this case, treatments are placed in only some blocks