Stat 430 - Design of Experiments   Lab 10 Nov 26 2012

Agenda
- The problem with saturated models
- The distribution of effect sizes
- Lenth's Method

The problem with saturated models

A saturated model is one in which there are as many parameters being estimated as there are sample units. A saturated model will always fit the data perfectly. However, since it fits the data perfectly; there is no way of determining how good the model will fit other similar data. In other words, it has no direct measure of uncertainty.

Example 1: N=1 for a single mean.

If you had a single response, y =5, and were using that to estimate the mean of something, what would your variance estimate look like?

```
y = 5
var(y)
```

This shouldn't be a surprize, because variance is estimated by…

… so we get an indeterminant answer. If we were to do a one-sample t-test on this data, we would get 0 degrees of freedom and can't get critical values or p-values. So R won't even try

```
t.test(y)
```

Likewise, if we expand this to a two-sample t-test but using two points, one in each sample:

```
y1 = 5
y2 = 10
t.test(y1,y2)
```

…it still won't work because we're trying to estimate the difference in two means, which involves estimating the two means. 2 parameters being estimated, 2 sample points, so 0 df, so no measure of uncertainty.

This is also true if only one of the two groups has a single sample size, because that group has no measure of uncertainty.

```
y1 = c(5,4,16,4,1,3,7,4,1,6)
y2 = 10
t.test(y1,y2)
```

Unless you let that group of size one 'borrow' an uncertainty measure by assuming equal variance.

```
t.test(y1,y2,var.equal=T)
```

This is also the case when the parameters are not means. Consider a simple linear regression with one slope and one intercept.

```
x = c(5,1)
y = c(44, 51)
mod = lm(y ~ x)
summary(mod)
```

This one has a geometric interpretation: A line can always be drawn between two points. Because both points are 'used up' in defining the line, there are no points left that are "free" to be anywhere else on the graph. In other words, there are no "degrees of freedom".

Finally, the same problem arises in ANOVA when each group is size 1.

```
y = c(5,1,53,15)
group = c("A","B","C","D")
mod = lm(y ~ group)
anova(mod)
```

By what do we care, the solution is simply to have some replication in your sample so that you have at least some degrees of freedom, right?   Not always possible.

The distribution of effect sizes

It's relatively easy to have pure replication in a sample for an experiment with only 2 or 3 factors at 2 or 3 levels each.   A 2^2 model only has 4 treatment combinations, 2^3 model has 8.

However, we're often interested in many factors, and each one doubles the sample units per replication, assuming there are only 2 levels which we're interested in. We can quickly run into a situation where replication is a luxury we can't afford.

Consider the following 2^4 design with N=16.   16 isn't an absurdly small sample size, but we still have no replication.

```
set.seed(100)
y = rnorm(16)   # Just 16 random points, no patterns
A = rep( c(-1,1), each = 1, times=8)
B = rep( c(-1,1), each = 2, times=4)
C = rep( c(-1,1), each = 4, times=2)
D = rep( c(-1,1), each = 8, times=1)
mod.24 = lm(y ~ A*B*C*D)
summary(mod.24)
```

We have estimates of the model coefficients (and through them of the effect sizes), but no idea yet how significant they are because we can't compare them to the level of random variability. What we can compare coefficients (and effect sizes) to is each other.

```
#Let's examine the distribution of the effect sizes
effects = 2 * mod.24$coef

# First, a histogram shows the effect sizes are symmetric and unimodel
hist(effects)

# A summary shows the mean of the effects to be near zero
summary(effects)

# Could they be normal?
# We can conduct the Shapiro-Wilks Test on this hypothesis:
shapiro.test(effects)

# We can also do a normal probability plot of the values
qqnorm(effects)
qqline(effects)


# And again with effect names
effect.names = names(mod.24$coef)
effect.x = qqnorm(effects,cex=0.1)$x
effect.y = qqnorm(effects,cex=0.1)$y
text(x=effect.x,y=effect.y,label=effect.names)
```

Why is this happening? Recall how each effect size is calculated: (Average of High) - (Average of Low), which is a sum of independent observed values.   If there is no such effect and what we're seeing is random variation, then these values are independent AND identically distributed, or iid. By the central limit theorem, the sum of iid values tends towards normality.

If there really is an effect, then the values at the high level for that factor are not identically distributed to the values at the low level for that factor. In that case, that particular effect size will NOT be normally distributed, and we can exploit that be identifying those effect sizes as outliers from a normal distribution.

One way to do that is with Lenth's Method

Lenth's Method (Section 6.5, Page 262)

Lenth's method is used to identify effects that are significant. That is, effects that are of a large enough size that we wouldn't expect to see them by random variation. It's used in cases of saturated models where we have no degrees of freedom left to estimate pure error.

In cases with 0 degrees of freedom, parametric methods fail us. So Length's method uses a non-parametric estimate of standard deviation of an effect size:

```
## Our data set: Example 6.2. Page 257
y = c(45,71,48,65,68,60,80,65,43,100,45,104,75,86,70,96)
A = rep( c(-1,1), each = 1, times=8)
B = rep( c(-1,1), each = 2, times=4)
C = rep( c(-1,1), each = 4, times=2)
D = rep( c(-1,1), each = 8, times=1)
ex.602 = data.frame(y,A,B,C,D)
mod.602 = lm(y ~ A*B*C*D,data=ex.602)
effects = 2 * mod.602$coef
effects = effects[-1]

# First, get a variance estimate of ALL the effect sizes
#1.5 times the median of the absolute value of the effect sizes (or contrasts, more
generally)
s0 = 1.5 * median( abs ( effects ) )

# Then, get the PSE, or PSUEDOSTANDARD ERROR.
# This is a non-parametric measure of the standard error
# It only uses values that are considered 'noise', and not outliers (outliers = more than
2.5*s0)

noise = which( abs(effects) < 2.5*s0)
pse = 1.5 * median( effects[noise])

## Now we have a (pseudo, meaning fake) measure of standard error.
## We can get a critical value for a significant value, we call this ME, or margin of error
## here it is calcuated for 95% confidence
m = length(effects) # total # of effects
d = m / 3
me =   - qt(0.025,d) * pse
```

```
## We can also adjust for multiple testing by using the SME, or simulataneous margin of error
## which is the same as ME, but with an adjustment to the critical-t to control experimentwise error
## Much like the Bonferroni, but with a different calculation
alpha.adj = 1 - (1 + 0.95^(1/m))/2
sme = -qt(alpha.adj,d) * pse
```

So if we're not adjusting for multiple testing, every effect of size greater than ME is considered significant. If we ARE adjusting for multiple testing, every effect of size greater than SME is considered signifcant.

```
which(effects > me)
which(effects > sme)
```

Note that really only works when there are a few significant effects among many, which is usually the case (principle of effect sparsity).