610 – R7 **Colleen F. Moore**
Psychology 610 **University of Wisconsin**

Designs with Nested Random Factors (between-groups) ANOVA in R (Equal or Balanced N only!!!)

This tutorial uses the data in "KeppelTable25.2top.xls".
The aov methods described here work ONLY for balanced designs (equal cell n's will give you a balanced design).

Contents of this handout:
Section 1: Balanced randomized groups design with one fixed variable, one random variable nested in the fixed variable, and individuals randomly assigned to levels of the fixed and random-nested variable.
Section 2: Balanced randomized groups design with two factorial fixed variables, a random variable nested in one of the fixed variables, and individuals randomly assigned to the cells.

Section 1.

I. Bring the data into R.

Type the data in plain text with a header row. My header row called the dv "recall" and the two factors are A and B. I labeled the factors numerically, rather than a1, a2 etc. Numerical labels can be risky because you might analyze them as if they were numbers instead of categories.

Important! Because the 3 levels of B are nested in the two levels of A, I labeled the variable called 'BnestinA' from 1 to 6 instead of from 1 to 3. (I also typed in the a variable called 'B' and labeled it from 1 to 3, but we can ignore B that for now.)

Paste the data to the clipboard. Then tell R this:

```
> your.data=read.table(pipe("pbpaste"),header=T)  #This creates a data frame called "your.data".
> your.data
```

```
   recall  A  B  BnestinA
1      14  1  1         1
2      15  1  1         1
3      12  1  1         1
4      13  1  1         1
5      13  1  1         1
6      13  1  2         2
7      12  1  2         2
8      15  1  2         2
9      16  1  2         2
10     12  1  2         2
11     15  1  3         3
12     17  1  3         3
13     15  1  3         3
14     16  1  3         3
15     15  1  3         3
```

```
16       7  2  1         4
17       8  2  1         4
18       7  2  1         4
19       7  2  1         4
20       8  2  1         4
21       8  2  2         5
22       6  2  2         5
23      11  2  2         5
24      11  2  2         5
25      11  2  2         5
26      13  2  3         6
27      10  2  3         6
28      11  2  3         6
29      11  2  3         6
30      10  2  3         6
```

```
> attach(your.data)  # there are pro's and con's of attaching data or not
> a=factor(A)  # make a factor called 'a'
> bnest=factor(BnestinA)  # make a factor called 'bnest'. It has 6 levels.
```

II. Do the anovas, and learn to control what R uses as the error term.

A. Wrong analysis. B nested, but ***not*** random. This uses the ***incorrect*** error term for testing factor A.

```
> wrong.aov=aov(recall~a*bnest)  # this will treat the S/ABcells as the error term.
```

Because I labeled the levels of bnest from 1 to 6, R will not try to make an interaction.

```
> summary(wrong.aov,intercept=T)
            Df Sum Sq Mean Sq  F value    Pr(>F)
(Intercept)  1 4130.1  4130.1 1982.464 < 2.2e-16 ***
a            1  182.5   182.5   87.616 1.759e-09 ***
bnest        4   47.3    11.8    5.680  0.002308 **
Residuals   24   50.0     2.1
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

B. Correct analysis by telling R the correct error term to use. The anova table has two sections, one that uses 'bnest' as the error, and one that uses 'within' as the error.

```
> nestedRand=aov(recall~a+Error(bnest))
> summary(nestedRand)

Error: bnest
          Df  Sum Sq Mean Sq F value   Pr(>F)
a          1 182.533 182.533  15.425 0.01714 *
Residuals  4  47.333  11.833
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
Df Sum Sq Mean Sq F value Pr(>F)
Residuals 24 50.000   2.083
```

*** Check that design is balanced!!!

> same.data=data.frame(recall,a,bnest) # make a new dataframe with just the pertinent variables in it, the dv and the factors

```
> !is.list(replications(recall~a+bnest, data=same.data)) # checks balance
[1] TRUE
> replications(recall~a+bnest, data=same.data) # check cell n's
   a bnest
  15     5
```

C. Incorrect analysis as if B were factorial to A.

```
> b=factor(B)
> wrongcrossed=aov(recall ~ a*b)
> summary(wrongcrossed, intercept=T)
            Df Sum Sq Mean Sq  F value    Pr(>F)    
(Intercept)  1 4130.1  4130.1 1982.464 < 2.2e-16 ***
a            1  182.5   182.5   87.616 1.759e-09 ***
b            2   42.9    21.4   10.288 0.0005934 ***
a:b          2    4.5     2.2    1.072 0.3581546    
Residuals   24   50.0     2.1                       
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note that the SS's for b and a:b add up to 'bnest' in the two analyses that structured the data with B nested in A.

III. Means, residuals, fitted values etc.

A. Means, sd's etc.

1. Use 'tapply'.

```
> tapply(recall, IND = list(a, bnest), mean)
     1    2    3   4   5  6
1 13.4 13.6 15.6  NA  NA NA
2   NA   NA   NA 7.4 9.4 11
> tapply(recall, IND = list(a, bnest), sd)
         1        2         3         4        5        6
1 1.140175 1.816590 0.8944272        NA       NA       NA
2       NA       NA        NA 0.5477226 2.302173 1.224745
# hmm, looks like the sd's are bouncing around a bit much.

> tapply(recall, IND=list(a,bnest),length)
   1  2  3  4  5  6
1  5  5  5 NA NA NA
2 NA NA NA  5  5  5
```

2. Use 'model.tables'.

```
> model.tables(nestedRand,"means",se=T)
Tables of means
Grand mean

11.73333
```

```
 a
a
     1      2
14.200  9.267
Warning message:
In model.tables.aovlist(nestedRand, "means", se = T) :
  SEs for type 'means' are not yet implemented
```

> model.tables(nestedRand,se=T) # when "means" isn't specified, it gives est effects. However, we get the est se by asking for the effects

```
Tables of effects

 a
a
      1       2
 2.4667 -2.4667

Standard errors of effects
             a
        0.8882
replic.     15
```

The estimated se is based on the MSerror for testing factor A, or the MS for B nested in A. Est se = sqrt(MSerror/number of obs in each mean) = sqrt(11.833/15). This is the "pooled" se.

We can use Baron & Li's standard error function to get estimated cell se's. These are not really of interest because b is a nested random factor.

```
> se <- function(x)
{  y <- x[!is.na(x)]   # remove the missing values
sqrt(var(as.vector(y))/length(y)) }
> serr=tapply(recall,IND=list(a,bnest),se); serr
```

```
        1         2   3         4        5         6
1 0.509902 0.8124038 0.4        NA       NA        NA
2       NA        NA  NA 0.2449490 1.029563 0.5477226
```

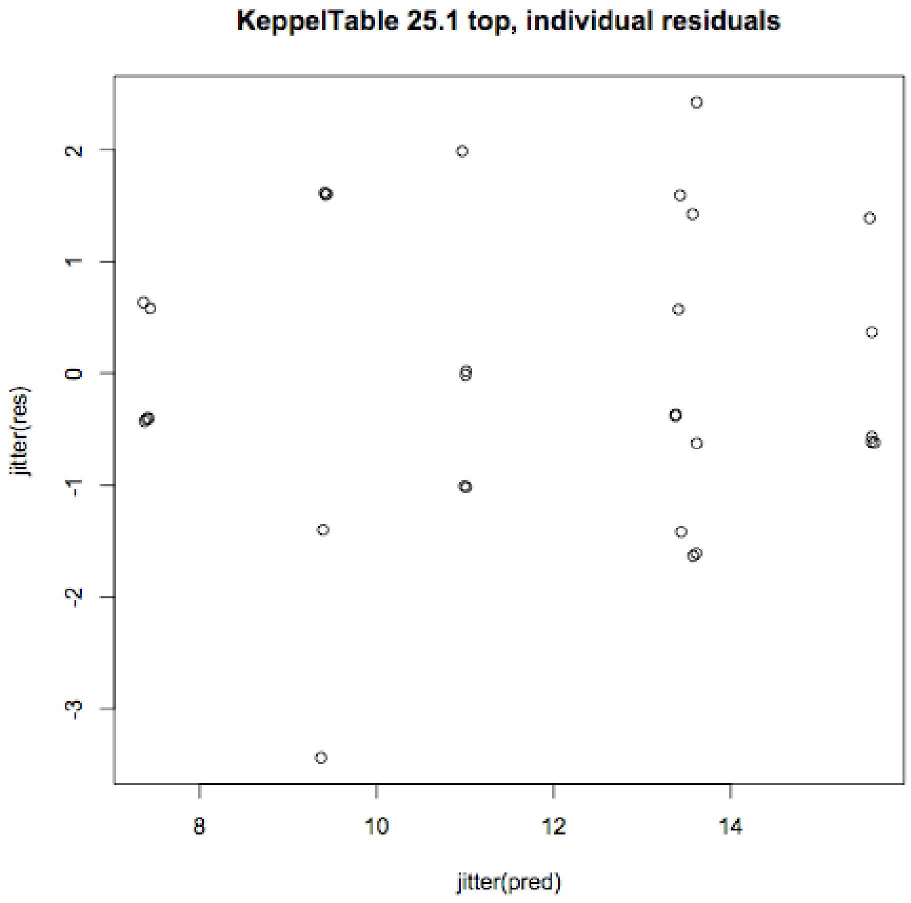
B. Check the model fit by graphing predicted and residuals.

To get the residuals we need to fit the model using 'lm' rather than 'aov'. I'm not sure why, but for a nested design 'aov' will not give pred and residuals.
Here are the statements to do the fit with 'lm'. Notice that 'lm' gives the incorrect F for factor A, as it does not use B-nested-in-A for error.

```
> lmnest=lm(recall~a+bnest)  # fit the linear model, put results in 'lmnest'
> anova(lmnest)   # get the anova table of the results
Analysis of Variance Table

Response: recall
          Df  Sum Sq Mean Sq F value    Pr(>F)
```

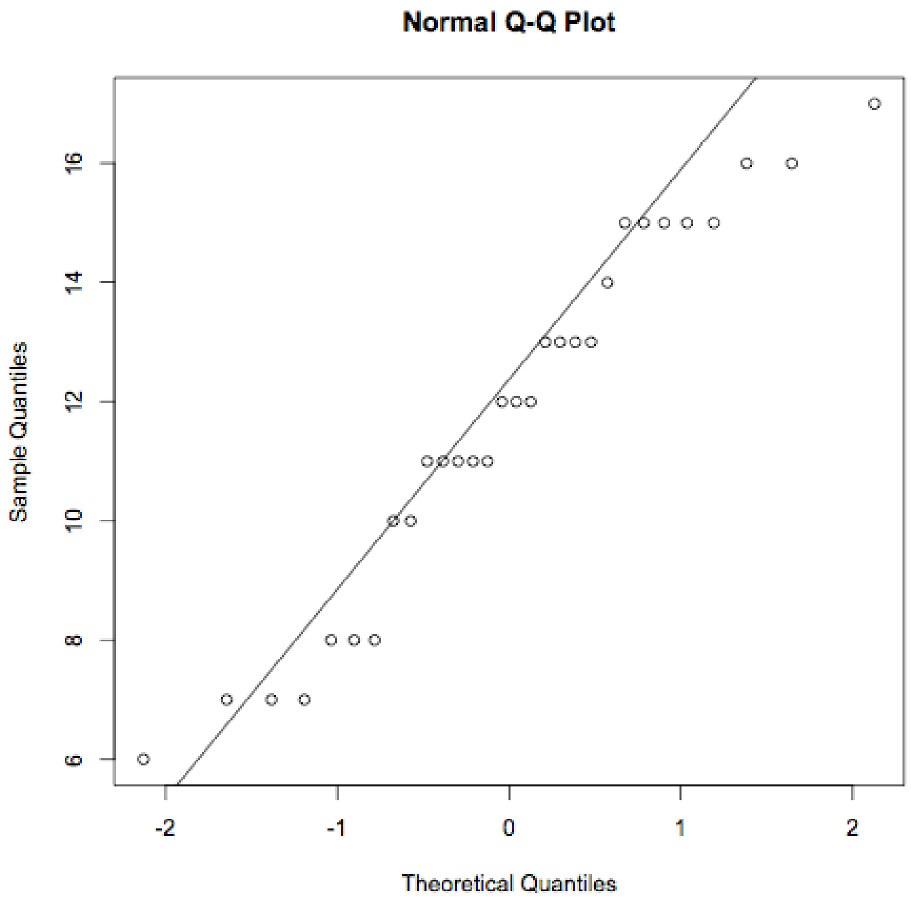
```
a           1 182.533 182.533  87.616 1.759e-09 ***
bnest       4  47.333  11.833   5.680  0.002308 **
Residuals 24  50.000   2.083
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
> pred=predict(lmnest)  # calc predicted values for the fit saved in 'lmnest'
> res=residuals(lmnest)  # calc residuals
> pred   # look at the actual numbers to see if they make sense.
   1    2    3    4    5    6    7    8    9   10   11   12   13   14
  15   16   17   18   19   20
13.4 13.4 13.4 13.4 13.4 13.6 13.6 13.6 13.6 13.6 15.6 15.6 15.6 15.6
15.6  7.4  7.4  7.4  7.4  7.4
  21   22   23   24   25   26   27   28   29   30
 9.4  9.4  9.4  9.4  9.4 11.0 11.0 11.0 11.0 11.0
```

> plot(jitter(pred),jitter(res),main="KeppelTable 25.1 top, individual residuals") # graph predicted vs residuals. Jitter in both directions so the individual points show better. This gives us the individual-level residuals.



IV. Check some assumptions
Graph the qq-normal plot. The points should lie on the line. The data are off in the tails.

```
> qqnorm(recall)
> qqline(recall)
```



Section 2.

Now we add another fixed variable that is factorial to a fixed variable. We have one random variable nested, in addition to replications (or individuals) who are randomly assigned to conditions. This section uses data in 'HO#23part2data.xls'. See Handout#23 for more info on the example.

I. Prepare data and bring into R. As in Section I, we label the nested variable, C, not 1 to 4, but 1 to 8 so that the confounding with factor B is clear.

```
> your.data=read.table(pipe("pbpaste"),header=T)
> your.data
   dv Asentence Bgender CnestedB
1   6         1       1        1
2   5         1       1        2
```

```
3   6          1        1         3
4   8          1        1         4
5   8          1        2         5
6   6          1        2         6
7   7          1        2         7
8   9          1        2         8
9   8          1        1         1
10  6          1        1         2
11  7          1        1         3
12  5          1        1         4
13  9          1        2         5
14  7          1        2         6
15  8          1        2         7
16  6          1        2         8
.
.
.
65  7          3        1         1
66  9          3        1         2
67  9          3        1         3
68 10          3        1         4
69  6          3        2         5
70  3          3        2         6
71  6          3        2         7
72  4          3        2         8
```

II. Do the anova, find means etc. The tricky part is **specifying the error**.

A. The anova

> attach(your.data) # you can use the $ notation instead of attaching
> Asent=factor(Asentence) # make numerical variables into factors
> Bgend=factor(Bgender)   # I always alter the var name when I make a factor. That way both the factora nd my original variable are distinct and I can transfer data easily to another program that wants numerical values for the factor levels.
> CnestB=factor(CnestedB)
> randNest2=aov(dv~Asent*Bgend+Error(CnestB/(Asent))) # we say that CnestB is the error for Asent. This means that R will use the CxA interaction as the error term for A, and will use C as the error for B. This is what we want. *** Always check what ANY anova program is using for the error term!
> summary(randNest2)

```
Error: CnestB
          Df  Sum Sq Mean Sq F value  Pr(>F)
Bgend      1 18.0000 18.0000  9.3462 0.02231 *
Residuals  6 11.5556  1.9259
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: CnestB:Asent
            Df Sum Sq Mean Sq F value    Pr(>F)
Asent        2   0.08    0.04  0.0206    0.9797
Asent:Bgend  2 326.08  163.04 80.5881 1.107e-07 ***
Residuals   12  24.28    2.02
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Error: Within
          Df  Sum Sq Mean Sq F value Pr(>F)
Residuals 48 128.000   2.667
```

# The results show a sig effect of Gender of the example, and a significant interaction of sentence form x gender. We'll graph these below.

Check that design is balanced!!
> same.data=data.frame(dv,Asent,Bgend,CnestB) # make a new data frame with the factors and dv in it
> !is.list(replications(dv~Asent*Bgend+Asent*CnestB, data=same.data)) # notice that I specified the model with AxB and AxC interactions only
[1] TRUE
> replications(dv~Asent*Bgend+Asent*CnestB, data=same.data) # check n's

```
   Asent        Bgend        CnestB  Asent:Bgend Asent:CnestB
     24           36            9           12            3
```

B. Means and standard errors

> model.tables(randNest2,se=T) # this gives estimated effects and se's

```
Tables of effects

 Asent
Asent
       1        2        3
-0.04167  0.04167  0.00000

 Bgend
Bgend
   1    2
-0.5  0.5

 Asent:Bgend
     Bgend
Asent 1       2
    1 -0.1250  0.1250
    2 -2.5417  2.5417
    3  2.6667 -2.6667

Standard errors of effects
         Asent  Bgend Asent:Bgend
        0.2903 0.2313      0.4106
replic.     24     36          12
```

# these estimated se's are based on MSerror for that effect divided by sqrt of the number of observations going into the means for that effect.

> model.tables(randNest2,"means")

```
Tables of means
```

```
Grand mean

6.5

 Asent
Asent
    1     2     3
6.458 6.542 6.500

 Bgend
Bgend
1 2
6 7

 Asent:Bgend
     Bgend
Asent 1     2
    1 5.833 7.083
    2 3.500 9.583
    3 8.667 4.333
```

Or using 'tapply' to get the two-way means, main effect means and est se's:

```
> tapply(dv, IND=list(Asent,Bgend),length)  # check cell n's
   1  2
1 12 12
2 12 12
3 12 12
> tapply(dv, IND=list(Asent,Bgend),mean)
         1        2
1 5.833333 7.083333
2 3.500000 9.583333
3 8.666667 4.333333
> tapply(dv, IND=list(Asent,Bgend),sd)
         1        2
1 1.527525 1.443376
2 1.507557 1.781640
3 1.669694 1.497473
> tapply(dv, IND=list(Asent),mean);tapply(dv, IND=list(Bgend),mean)
       1        2        3
6.458333 6.541667 6.500000

1 2
6 7
```

> tapply(dv, IND=list(Asent),se) # using Baron & Li's se function above we can get estimated se's at any level of the design. HOWEVER, these main effect estimated se's do not take into account the increase in variability due to the presence of the other factor in the design. Therefore, I prefer to use the model-based se's for graphing the main effect means.

```
        1         2         3
0.3240324 0.7146986 0.5516773
```

> tapply(dv, IND=list(Bgend),se) # See comment above on Asent se's.

```
        1         2 
0.4382505 0.4436501 
```

> tapply(dv, IND=list(Asent,Bgend),se) # these cell sd/sqrt(cell n). Notice they are all larger than the model-based estimate of .4106 given by the 'model.tables' statement. This illustrates that using 'residuals' in a study with a random factor is not quite right.

```
          1         2
1 0.4409586 0.4166667
2 0.4351941 0.5143153
3 0.4819992 0.4322831
```

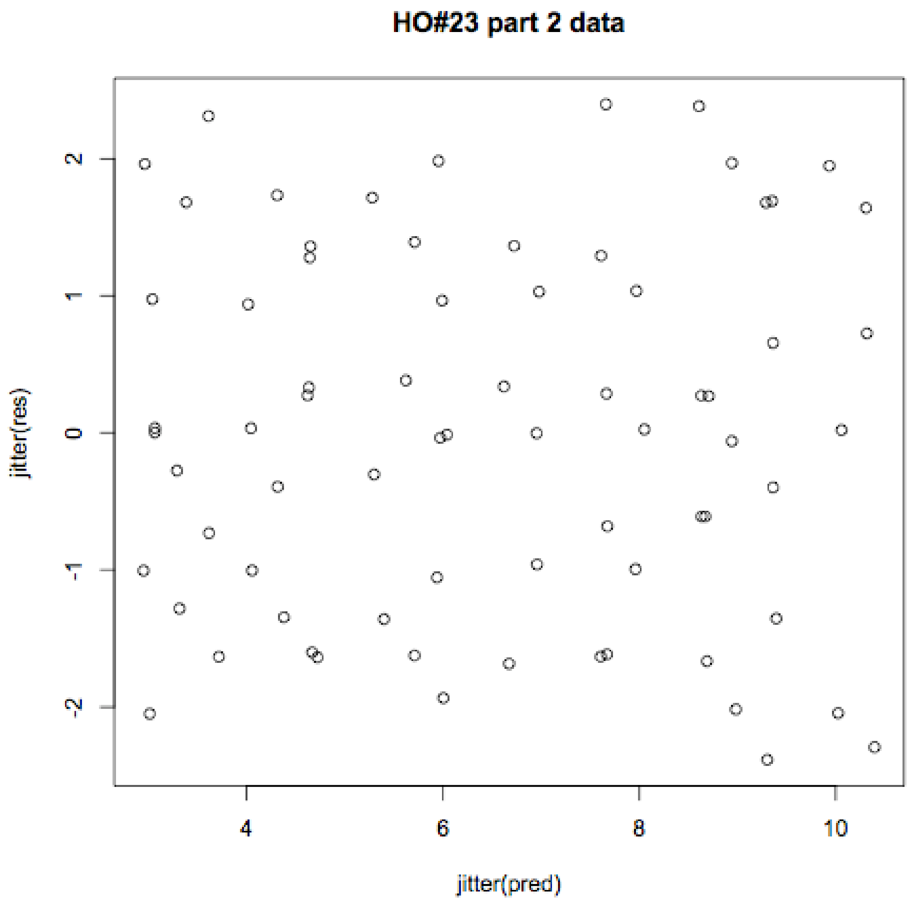
C. Find and plot predicted and residual values

R will not calculate predicted values from 'aov' for nested designs, so first we fit the model with 'lm'. I specify the AxB and AxC interaction, but not the BxC interaction because C is nested in B.

> lmrandnest = lm(dv~Asent*Bgend+CnestB*Asent)
> anova(lmrandnest)

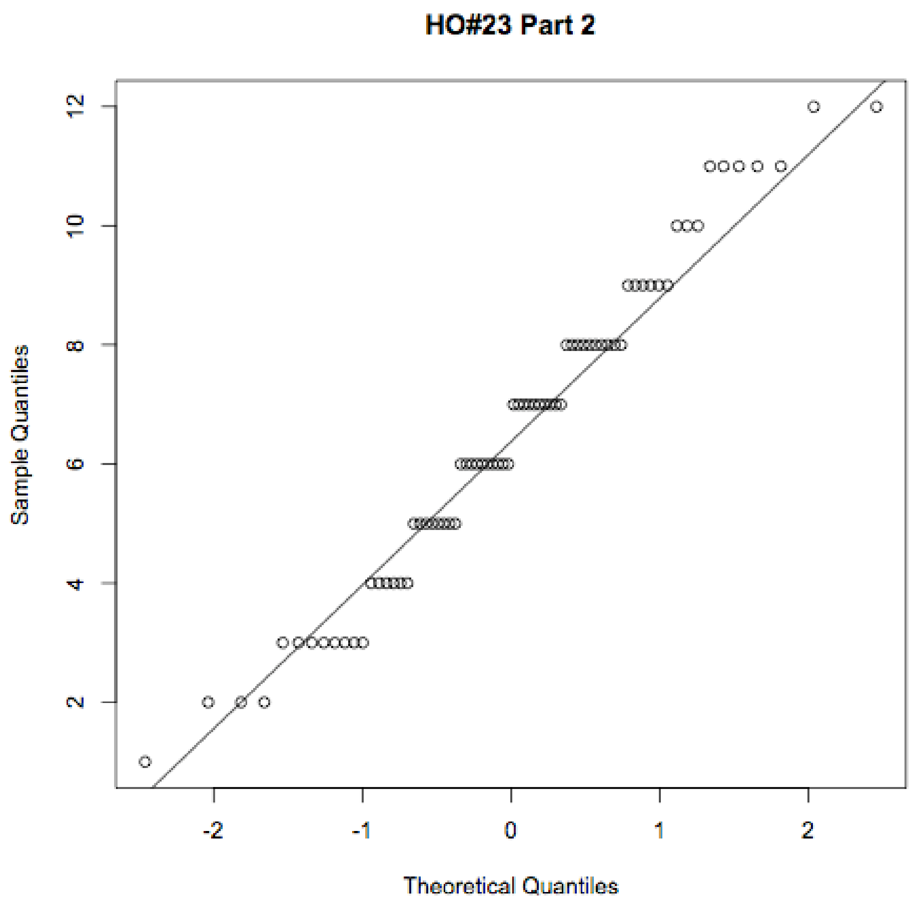
```
Analysis of Variance Table

Response: dv
             Df Sum Sq Mean Sq F value    Pr(>F)    
Asent         2   0.08    0.04  0.0156   0.98450    
Bgend         1  18.00   18.00  6.7500   0.01241 *  
CnestB        6  11.56    1.93  0.7222   0.63372    
Asent:Bgend   2 326.08  163.04 61.1406 6.336e-14 ***
Asent:CnestB 12  24.28    2.02  0.7587   0.68808    
Residuals    48 128.00    2.67                      
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

# the SS's are the same, but 'lm' uses the overall residual as the error term for everything.
> res=residuals(lmrandnest) # calculate the residuals
> pred=predict(lmrandnest)



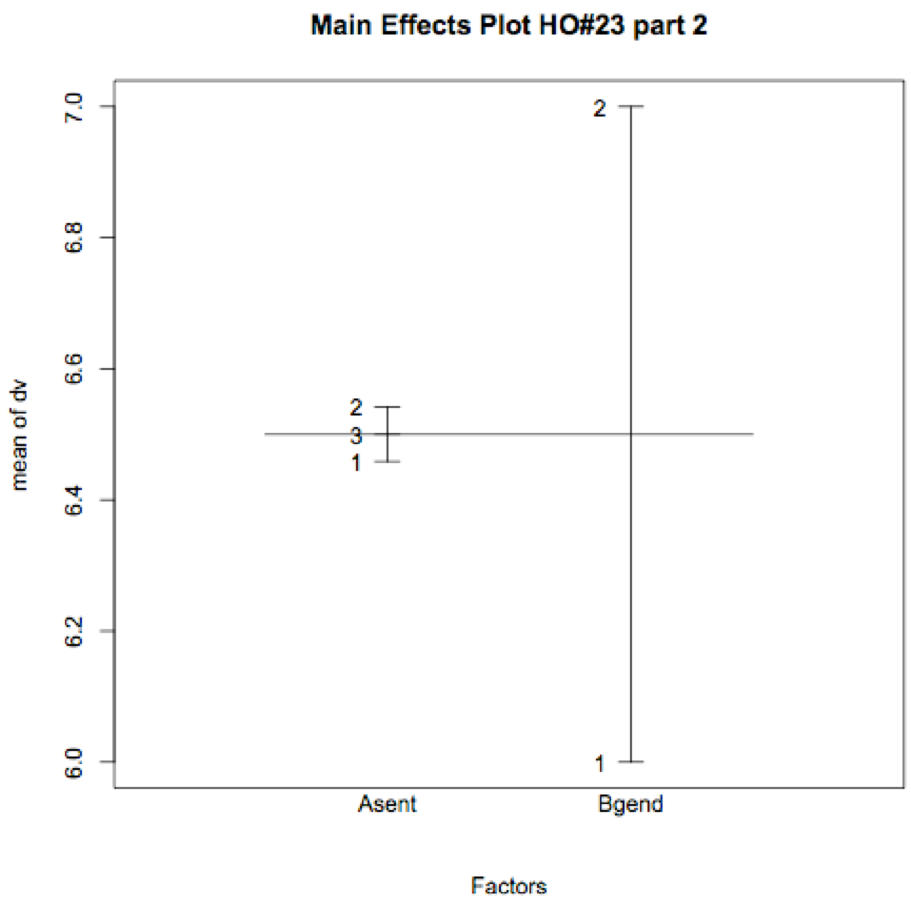
# The residuals x pred graphs appears pretty random via eyeball.

Normal qq plot:



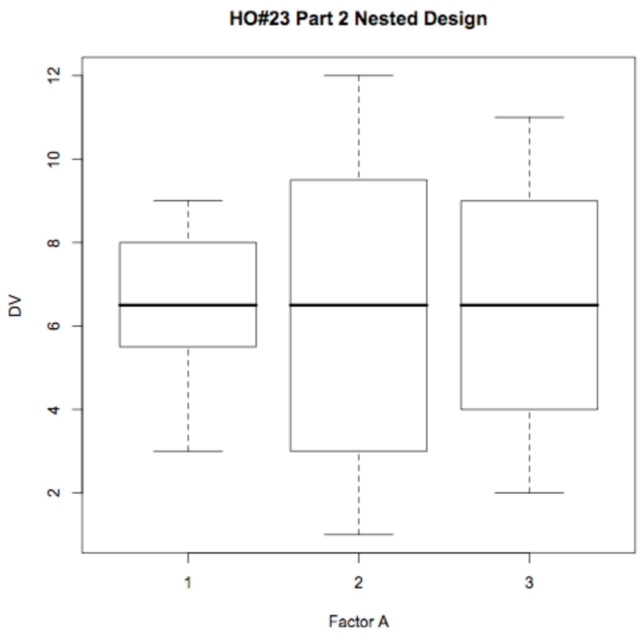
D. Graph the main effects and interactions
1. Factor A main effects graph:

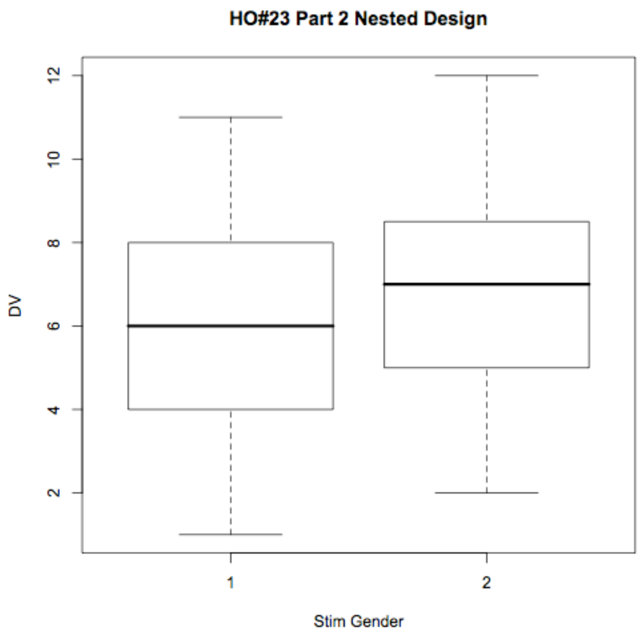
```
> plot.design(dv~Asent*Bgend,data="your.data",main="Main Effects Plot HO#23 part 2")
```
this shows that Gender of stimulus is having a large main effect compared to Sentence type.

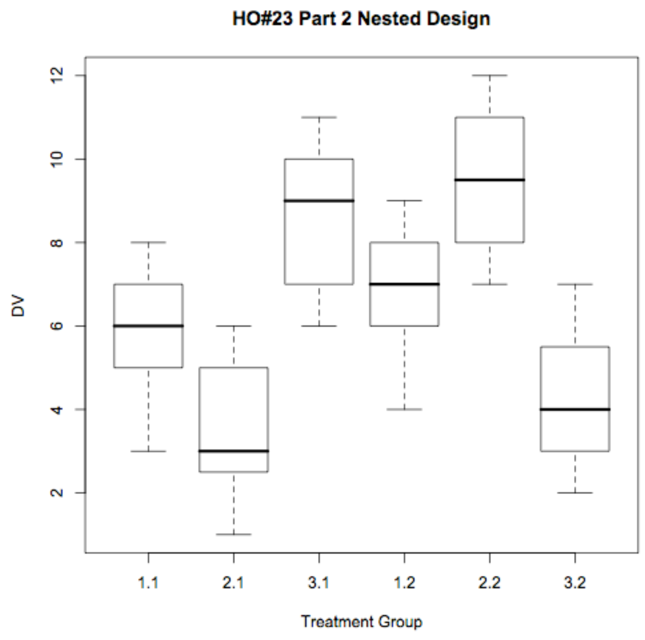


2. Boxplots are quick and easy:

```
> boxplot(dv~Asent,main="HO#23 Part 2 Nested Design", xlab="Factor A",ylab="DV")
> boxplot(dv~Bgend,main="HO#23 Part 2 Nested Design", xlab="Stim
Gender",ylab="DV")
> boxplot(dv~Asent:Bgend,main="HO#23 Part 2 Nested Design", xlab="Treatment
Group",ylab="DV")
```

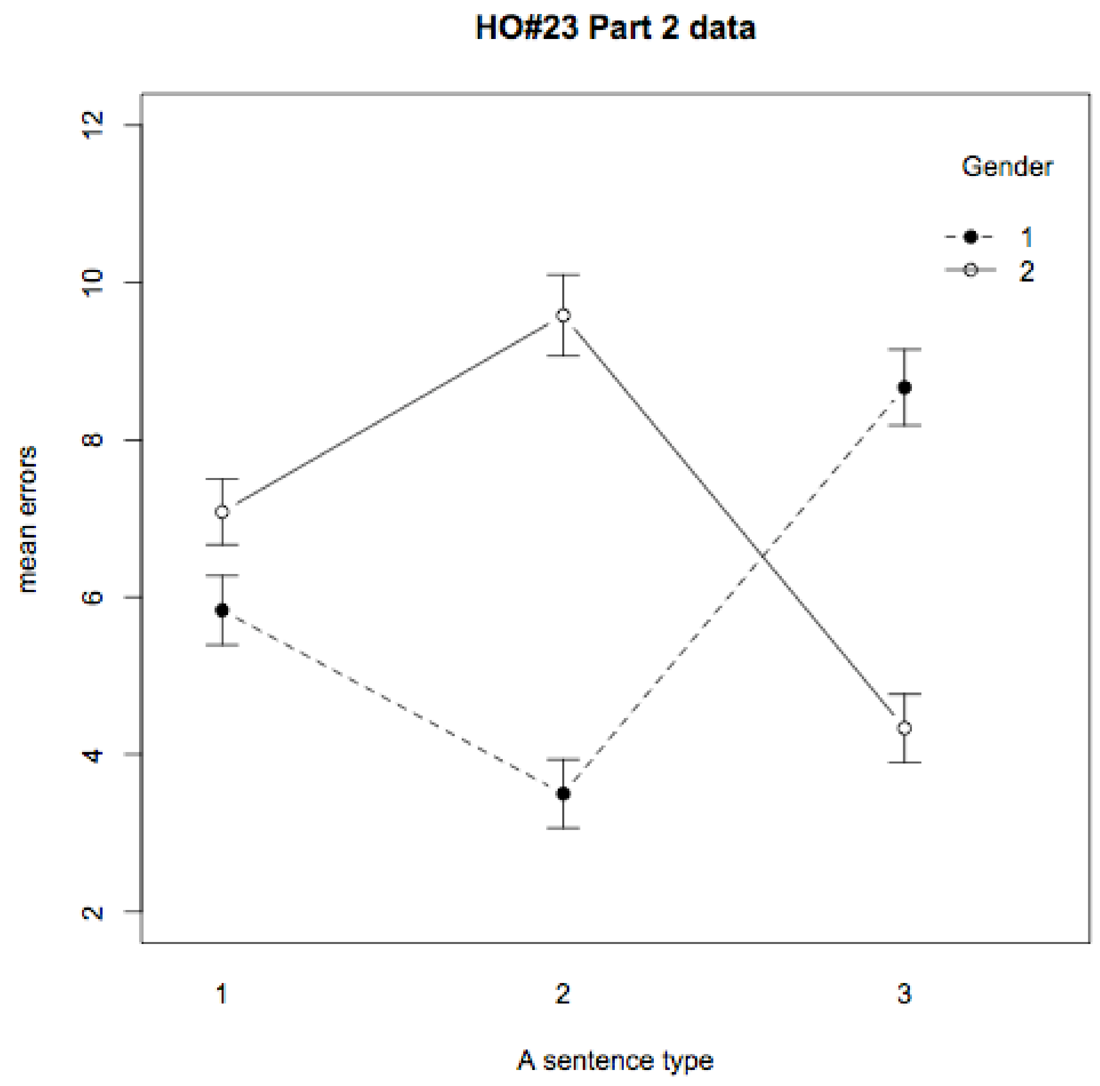

HO#23 Part 2 Nested Design
DV
2
4
6
8
10
12
1
2
3
Factor A


HO#23 Part 2 Nested Design
DV
2
4
6
8
10
12
1
2
Stim Gender



3. Interaction plot

```
> library(sciplot)  # activate sciplot in order to use 'lineplot.CI'

> lineplot.CI(Asent, dv, group=Bgend , type="b",  legend=TRUE, trace.label="Gender",
leg.lab=NULL, fixed=FALSE, x.leg=NULL,  y.leg=NULL, cex.leg=1, ncol=1, xlab="A
sentence type",ylab="mean errors", pch=c(16, 21, 15, 22, 17, 24, c(3:14)),  xlim=NULL,
ylim=c(2,12), cex=NULL, lwd=NULL, col="black", cex.axis=1,  xaxt="s", data=NULL,
subset=NULL, main="HO#23 Part 2 data")  #
```



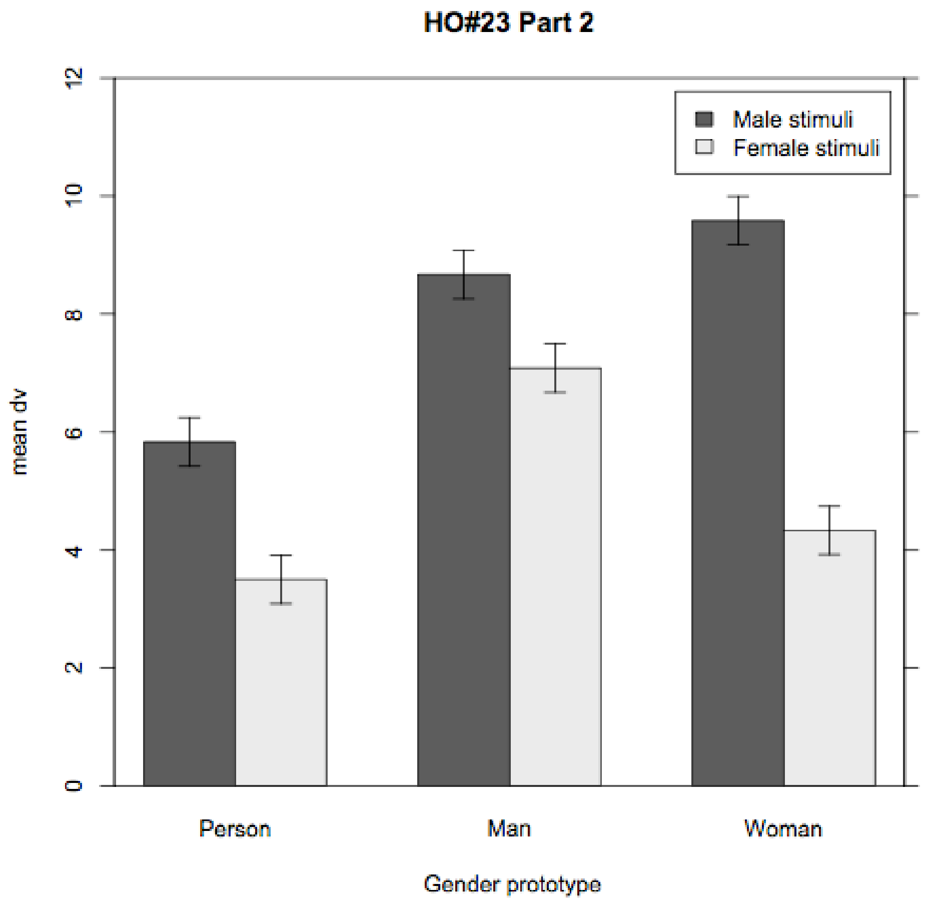
Now a bar graph:
Enter the function "superpose". We'll use it to make the error bars:

```
> superpose.eb <-
 function (x, y, ebl, ebu = ebl, length = 0.08, ...)
    arrows(x, y + ebu, x, y - ebl, angle = 90, code = 3,
    length = length, ...)
```

Bar graph of the interaction. I built this from scratch by entering the means and the model se's into matrices. That way I can use the model-based se's in the graph.

```
> dvmatrix=matrix(c(5.833,3.5,8.667,7.0833,9.5833,4.333),2,3) # make a matrix of means
> sematrix=matrix(c(.4106,.4106,.4106,.4106,.4106,.4106),2,3)  #model standard error est for each mean
> colnames(dvmatrix)=c("Person","Man","Woman")  # to get labels on the graph for the levels of the factor, enter row and column names for the matrix containing the means
> rownames(dvmatrix)=c("Male stimuli","Female stimuli")
> dvmatrix   # show results and check see if I did this the way I wanted to
               Person    Man  Woman
Male stimuli    5.833 8.6670 9.5833
Female stimuli  3.500 7.0833 4.3330
```

```
> dvplot=barplot(dvmatrix,beside=T,ylim=c(0,12),main="HO#23 Part 2",xlab=" Gender prototype", ylab="mean dv", axes=T, legend.text=T, offset=0, xpd=F)  #
> box()  # put a box around the plot
> axis(4,labels=F)    # add tick marks on the right side
> superpose.eb(dvplot,dvmatrix,sematrix,col="black",lwd=1)  # use the 'superpose' error bar function with the values in 'sematrix'
```


HO#23 Part 2
mean dv
■ Male stimuli
□ Female stimuli
Person
Man
Woman
Gender prototype