Psych 610 Handout – R3
Post-Hoc Tests and Effect Size
One-Way Between Groups Designs

Colleen Moore & Mike Amato
University of Wisconsin - Madison

This handout uses data from Myers, Fundamentals of Experimental Design, 2nd Ed, p. 109. The study has 6 groups. The data are on the 610 course website in a file named "MyersDataOneWay.txt".

1. Bring in the data

An alternative to typing the full name of your data file is to navigate to it with the following. Note this does **not** change your working directory, so be careful if you save your workspace or any graphs.

```
> d1 = read.table(file.choose(), header=T)   # opens a window for file navigation
```

Check your data to make sure it read in correctly. Once you're satisfied, that it has tell R that *group* is a categorical factor.

```
> d1$group = as.factor(d1$group)
```

OPTIONAL: Instead of just typing the name of the data frame, and getting 48 unwieldy lines of output, we can use a *for* loop to create a tidy display for ourselves. First we define a null object, arbitrarily calling it "TidyDisplay". Then we iterate over the numbers 1 through 6, since we have 6 groups. At each iteration we use *subset( )* to make a vector of scores for that group, then *cbind( )* adds that column onto TidyDisplay.

```
> TidyDisplay = NULL
> for (i in 1:6)  TidyDisplay = cbind(TidyDisplay, subset(d1$score, d1$group==i))
> TidyDisplay
     [,1] [,2] [,3] [,4] [,5] [,6]      #scores for each group are shown in columns
[1,]    7    6    9   42   28   13
[2,]   33   11   12   25    6   18
[3,]   26   11    6    8    1   23
[4,]   27   18   24   28   15    1
[5,]   21   14    7   30    9    3
[6,]    6   18   10   22   15    4
[7,]   14   19    1   17    2    6
[8,]   19   14   10   32   37    2
```

2. Do the omnibus Anova

Test the null hypothesis that at least two groups have different means from each other.

```
> aov.omni = aov(score~group, data=d1)
> summary(aov.omni, intercept=T)
              Df   Sum Sq  Mean Sq   F value     Pr(>F)
(Intercept)    1  11102.1  11102.1  135.8251  9.586e-15 ***
group          5   1554.9    311.0    3.8046   0.006228 **
Residuals     42   3433.0     81.7
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Ok, looks like we have a significant effect of group. Now let's explore it a little more deeply, looking at its overall effect size, as well as some pairwise comparisons.

3. Calculate the effect size of group

In the omnibus ANOVA, we found that group does indeed explain a significant amount of variance. But how much? Is this a small, medium, or large effect size?

Manually calculate η^2, also known as R^2, also known as "percent variance explained."

```
> 1555.9/(1554.9+3433.0)
[1] 0.3119349
```

Now ask R to show it to us. Happily the numbers are similar, although slightly off due to some rounding error.

```
> summary(aov.omni.lm)$r.squared      # r.squared is an attribute of the summary object
[1] 0.3117367
```

Recall that η^2 measures the effect size in our sample, but what we are really interested in is the estimated effect size in the population, which we know will be smaller than the observed effect size in our sample. As of this writing, Mike is unsure how to return it in R. Fortunately we can calculate it easily enough with the following formula from Keppel:

$$\boxed{\text{estimated } \omega^2 = \hat{\omega}^2 = \frac{(a-1)(F-1)}{(a-1)(F-1) + an}}$$

```
> OmegaSquaredHat = (6-1)*(3.8046-1)/( (6-1)*(3.8046-1)+6*8 )
> OmegaSquaredHat
[1] 0.2260935
```

3A. Test all pairwise mean differences as planned comparisons (i.e., unadjusted)

This is easy to do in R. The following command gives a p-value for each comparison.

```
> PairwiseUnadjusted = pairwise.t.test(d1$score, d1$group, p.adjust="none")
> PairwiseUnadjusted
        Pairwise comparisons using t tests with pooled SD
data:  d1$score and d1$group

  1       2       3       4       5
2 0.25204 -       -       -       -
3 0.04702 0.38127 -       -       -
4 0.16583 0.01375 0.00127 -       -
5 0.27499 0.95616 0.35251 0.01577 -
6 0.02679 0.26334 0.80468 0.00061 0.24110

P value adjustment method: none
```

Family-wise alpha

$$\alpha_{FW} = 1 - (1-\alpha)^{\#\text{ tests}}$$

If we don’t make any adjustments to our p-values, we find that 6 of the 15 differences appear significant. For example, the mean of group 2 is significantly different from the mean of group 4, p=.01375.

However by conducting 15 tests at $\alpha = .05$, we actually end up with a family-wise $\alpha = 1-(1-.05)^{15} = .54$!!!

3B. Test all pairwise differences while controlling for family-wise type I error

```
> pairwise.t.test(d1$score, d1$group, p.adjust="bonferroni")
> pairwise.t.test(d1$score, d1$group, p.adjust="holm")
```

I won't waste paper showing the output here, but compare the results of the two methods. Note that Holm's is more powerful than Bonferroni's. For this reason, the Holm Method is the default for this function.

Alternatively to using those methods for controlling family-wise error, you could control the False Discovery Rate by setting *p.adjust="fdr"* .

4. Tukey Test

First we'll manually calculate a p-value and confidence interval to test the difference between the means of group1 and group2, controlling α $_{FW}$ using the Tukey method. We can get all the information we need from a *tapply( )* command, and the summary of the omnibus ANOVA.

```
> summary(aov.omni, intercept=T)
            Df  Sum Sq Mean Sq  F value    Pr(>F)    
(Intercept)  1 11102.1 11102.1 135.8251 9.586e-15 ***
group        5  1554.9   311.0   3.8046  0.006228 ** 
Residuals   42  3433.0    81.7                       
> groupmeans = tapply(d1$score, d1$group, mean)
> numgroups = 6
> n = 8
> dferror = 42
> mserror = 81.7
> coeff.1v2 = c(1,-1,0,0,0,0)        # contrast coefficients to test group1 vs. group2
> psi = sum(coeff.1v2*groupmeans)         # this is just the difference between the observed means
> psi
[1] 5.25
```

As always, since we're going to use the observed difference in our sample as an estimate of the true difference in the population, we need to know how good an estimate it is. For that we'll calculate the standard error of the difference. Note that it is the same as the standard error of the mean.

```
> sediff = sqrt( mserror/n )
> sediff
[1] 3.1957
```

Calculate the test statistic q, the studentized range statistic

```
> calcq = psi/sediff
> calcq

[1] 1.642832
```

Compare the value of calcq to the table value of the studentized range statistic for 6 groups and dferror = 42.

```
> q.1v2 = qtukey(.95, numgroups, dferror)
[1] 4.221779
```

Since our calculated q of 1.64 is less than table q of 4.22, we retain the null hypothesis. We can ask for a p-value, showing us the likelihood that the observed difference was due to chance.

```
> p.1v2 = ptukey(calcq, numgroups, dferror, nranges=1, lower.tail=F)
[1] 0.8521682
```

Lastly we can calculate a confidence interval around the difference. Since we know the test turned out non-significant, the confidence interval should cross zero.

```
> lowerbound = psi - q.1v2 * sediff
> upperbound = psi + q.1v2 * sediff
> lowerbound; upperbound
[1] -8.241541
[1] 18.74154
```

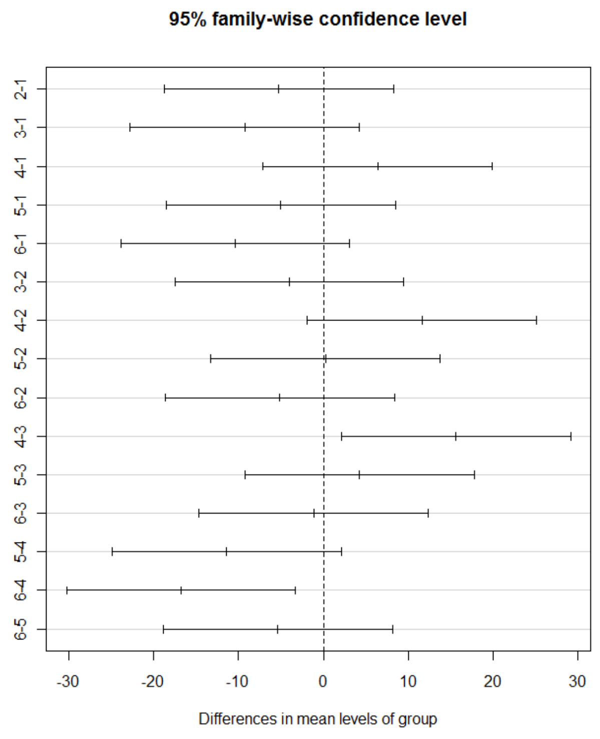
If you don't feel like conducting the rest of the pairwise Tukey tests manually, you can have R do them for you. The *TukeyHSD( )* function works similar to the *summary( )* function, in that it takes an ANOVA model as an argument.

```
> TukeyHSD(aov.omni, conf.level=.95)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = score ~ group, data = d1)

$group
       diff        lwr       upr     p adj
2-1  -5.250 -18.744686  8.244686 0.8522904
3-1  -9.250 -22.744686  4.244686 0.3346360
4-1   6.375  -7.119686 19.869686 0.7207042
5-1  -5.000 -18.494686  8.494686 0.8761170
6-1 -10.375 -23.869686  3.119686 0.2188268
3-2  -4.000 -17.494686  9.494686 0.9480751
4-2  11.625  -1.869686 25.119686 0.1271135
5-2   0.250 -13.244686 13.744686 0.9999999
6-2  -5.125 -18.619686  8.369686 0.8644848
4-3  15.625   2.130314 29.119686 0.0149617
5-3   4.250  -9.244686 17.744686 0.9336134
6-3  -1.125 -14.619686 12.369686 0.9998597
5-4 -11.375 -24.869686  2.119686 0.1424939
6-4 -16.750 -30.244686 -3.255314 0.0075302
6-5  -5.375 -18.869686  8.119686 0.8395496
```

Comfortingly, the values we calculated for the difference of group 1 vs. 2 match up with the values R calculates. They are off slightly due to rounding error. The signs are flipped on the bounds of the confidence interval, but it doesn't matter.

We can plot the confidence intervals for each test, to easily see which groups reliably differ. Depending on the resolution of your monitor, you may need to stretch the graph vertically to see all the labels on the y-axis.

```
> plot( TukeyHSD(aov.omni, conf.level=.95)
```



4B. Scheffe method programmed

Note: This page was copied from a previous version of this handout. The code works, but refers to a few variables that have not actually been created if you have been following along with the following pages. They include the following, as well as possibly others:

> ***nn** is a vector of the n in each group, got with tapply(score, group, length)*
> ***mm** is a vector of the means in each group, got with tapply(score, group, mean)*
> ***semicolons** separate 2 commands on a single line of code.*

Remember, Scheffe is best to use for complex contrasts, or when you are doing a lot of contrasts. We have yet to find a Scheffe method that anyone else has built for us in R, though one is almost certainly out there somewhere.

```
> dfnum=5; dferror=42; mserror=81.74       # enter values of df and mserror

> coeff=c(5,-1,-1,-1,-1,-1)                # this contrast compares the first group to the other 5
> psi=sum(coeff*mm); psi                   # calculate psi

[1] 23.5

> sscoeff=sum(coeff*coeff/nn)              # calculate the sum of the squared contrast coefficients
> mspsi=(psi*psi)/sscoeff; mspsi           # find the mean squared for the contrast
[1] 147.2667
> contrastF=mspsi/mserror; contrastF     # calculate the F for the contrast
[1] 1.801648
> tableF=qf(.05,dfnum,dferror,lower.tail=F)      # find the table F
[1] 2.437693
> critvalue=dfnum*tableF; critvalue # convert tableF to ScheffeF by multiplying by numerator df
[1] 12.18846
```

So comparing the contrast F to the critical value, we find that this contrast is nonsignificant by the Scheffe method. We can make Scheffe confidence intervals:

```
> lowbound=psi-sqrt(critvalue)*sqrt(mserror*sscoeff)
> upperbound=psi+ sqrt(critvalue)*sqrt(mserror*sscoeff)
> confinterval=c(lowbound,upperbound); confinterval
[1] -37.62339  84.62339                    # this would be useful for a graph
```

4C. Fisher-Hayter method
Similar to the Tukey contrast above, just find the qtukey and ptukey with dfnum instead of number of groups:

```
>  qtukey(.95,dfnum,dferror)
[1] 4.030232
> ptukey(calcq,dfnum,dferror,nranges=1,lower.tail=F)
[1] 0.77272
```

No matter how we slice it, there is just no significant difference between group1 and group2.