# Complete Example (Repeated Measures With Interactions)

New to this section:

* With repeated measures designs, you have to restructure the data into long format after data screening. The melt function creates one IV and one DV column, which includes the *conditions* in a repeated measures design.
* While it is very tempting to just use it as one IV, you would be ignoring the interaction component of the two IVs. Therefore, you have to create two new IV columns to represent the separate IV levels.
* Main effects: interpreting each IV on its own, ignoring the effect of the other IVs.
* Interaction: interpreting the IVs together, seeing if the conditions are significantly different OR if the pattern of data across levels is different for the other IV.

Chart of ANOVA Analysis:

|  |  |  |  |
| --- | --- | --- | --- |
|  | ANOVA | | |
|  | Main Effect | Main Effect | Interaction |
| If levels > 2  And significant | Dependent t-test  Bonferroni correction | Dependent t-test  Bonferroni correction | SPLIT one IV column  Dependent t-test  Bonferroni correction |
| If levels = 2 | Interpret means | Interpret means |

If the interaction is significant, often people ignore any analyses with the main effects:

* This procedure reduces Type 1 error because you are running less post hoc tests.
* You are interested in the interaction anyway, so why only interpret one variable at a time?

In this experiment people were given word pairs to rate based on their “relatedness”. How many people out of a 100 would put LOST-FOUND together?

**Datafile:** rm 2 anova.csv

**IVs:**

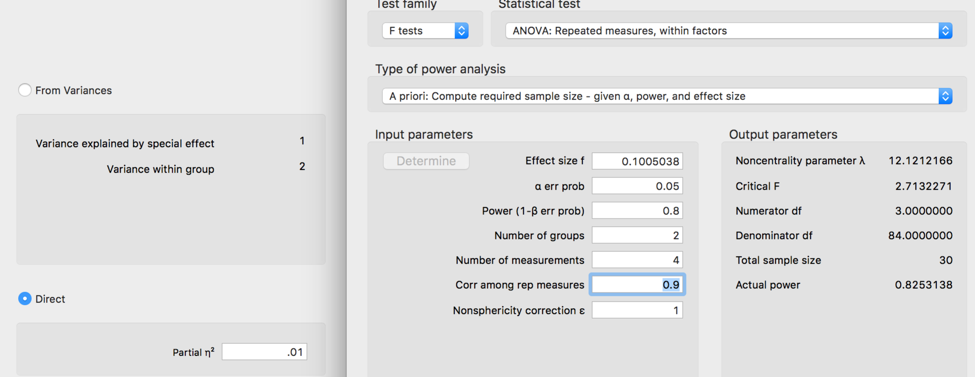
* FSG – how strong the relationship was between LOST-FOUND
  + FSG – two levels (low versus high)
* BSG – how strong the relationship was between FOUND-LOST
  + BSG – two levels (low versus high)
* Creates 4 conditions (lo lo, lo hi, hi lo, hi hi).
* Everybody got all four types of word pairs.

**DV:**

* Rating of each word pair type.

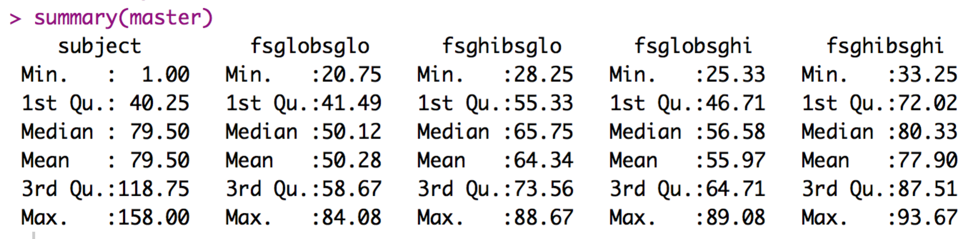
**Power:**

1. Open Gpower!
   1. Test family: F-test
   2. Statistical Test: ANOVA repeated measures, within factors
   3. Estimate an effect size: click determine 🡪 click direct 🡪 use eta square sizes you think might be accurate, remember small, medium, and large estimates from the notes.
   4. Alpha = .05
   5. Power (1-beta .20) = .80
   6. Number of groups = number of IVs
   7. Number of measurements = number of levels or conditions
   8. Corr among rep measures = correlation between levels or conditions
      1. You can estimate from previous research.
      2. Look at the correlations in a pilot study, go with the lowest one you find.
      3. .5-.7 is a good estimate if you are giving them the same test a couple times.
   9. Nonsphericity correction = epsilon
      1. You will not really know this number before you start a study. More useful if you have some participants to estimate from (see below on how to get that number).
2. Let’s estimate the following:
   1. Small effect size
   2. Two IVs
   3. Conditions from our current study
   4. Correlation = .9
   5. Epsilon = 1
3. Says we needed to run 30 people to find a significant effect with a small effect size.

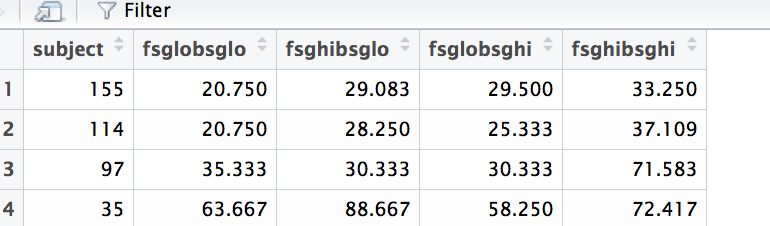


**Assumptions:**

1. Accuracy:
   1. Use the summary(*dataset name*) function to get the basic information for the data.
   2. Let’s check out minimum and maximum:
      1. This data should be rated 0-100.
      2. Just looking at the condition columns, we are ok because nothing is below zero or over 100.



1. Missing:
   1. With the summary function, I can also see that I don’t have any missing data, because there are no NA values shown. Therefore, I can skip the missing data step.
   2. Even if there was missing data, remember that any missing data ends up being more than 5% for each participant in an ANOVA. Therefore, they should normally get excluded.
2. Outliers:
   1. This data set is currently in WIDE format. What does that mean? It means that each person gets their own row, with each level as a different column.

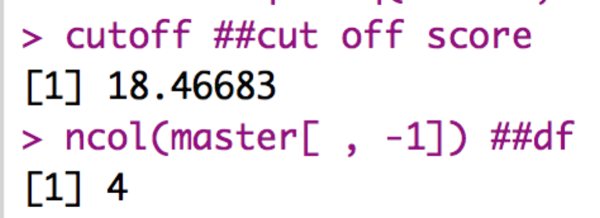


* 1. Because they are in this format, we have several columns to work with, which means we can use Mahalanobis values. We want to use this format for data screening because it accounts for the fact that people have more than one measurement. We would not want to ignore that person one is person one for all three levels.
  2. Create the Mahalanobis values:
     1. mahal = mahalanobis(*dataset*,

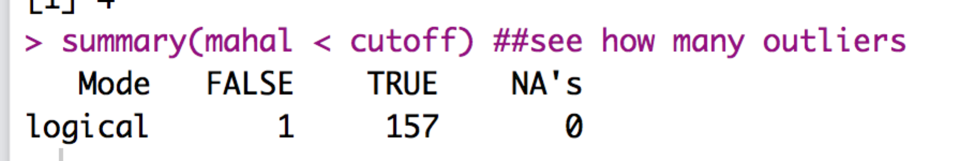
colMeans(*dataset*, na.rm = T),

cov(*dataset*, use = “pairwise.complete.obs”))

* 1. Create the cut off score:
     1. cutoff = qchisq(1-.001, ncol(*dataset*))
  2. Remember you can use:
     1. cutoff to get the cutoff score
     2. ncol(*dataset*) to get the *df*

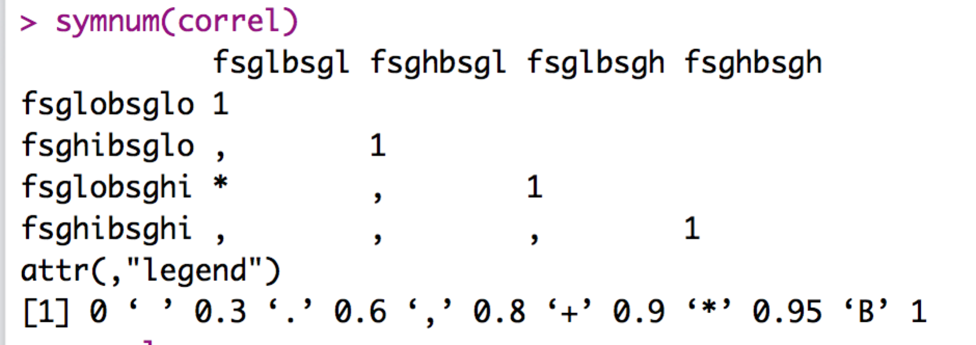


* 1. See how many outliers you have:
     1. summary(mahal < cutoff)

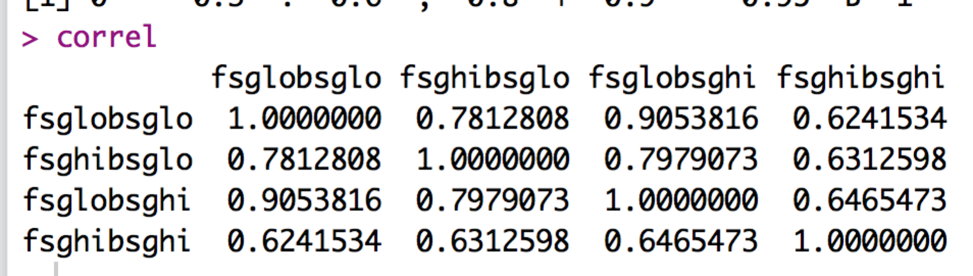


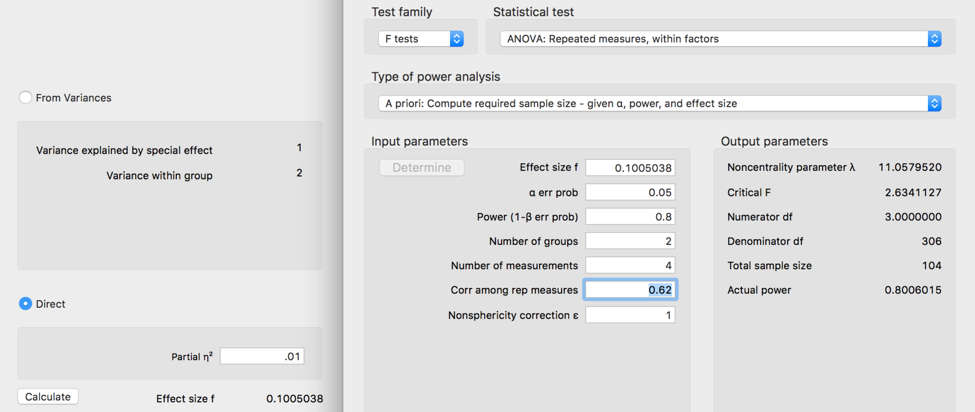
* + 1. Remember FALSE is bad.
    2. I have one outlier!
  1. Exclude outliers:
     1. noout = subset(*dataset*, mahal < cutoff)

1. Additivity
   1. We do have to worry about correlations in a repeated measures design, but not quite in the same way we talked about it for overall data screening.
   2. In general, you *want* the various measurements to be highly correlated – it will give you more power if they are correlated and less if they are not.
   3. However, they cannot be perfectly correlated or the ANOVA will not run.
   4. Mainly we are checking that we don’t get any 1s other than the diagonal in our symbols chart. So, basically, the rule is the *r* < .999.
   5. Get the correlations:
      1. correl = cor(*dataset*, use = “pairwise.complete.obs”)
   6. Get the symbols chart:
      1. symnum(correl)
   7. Look for 1s NOT on the diagonal:

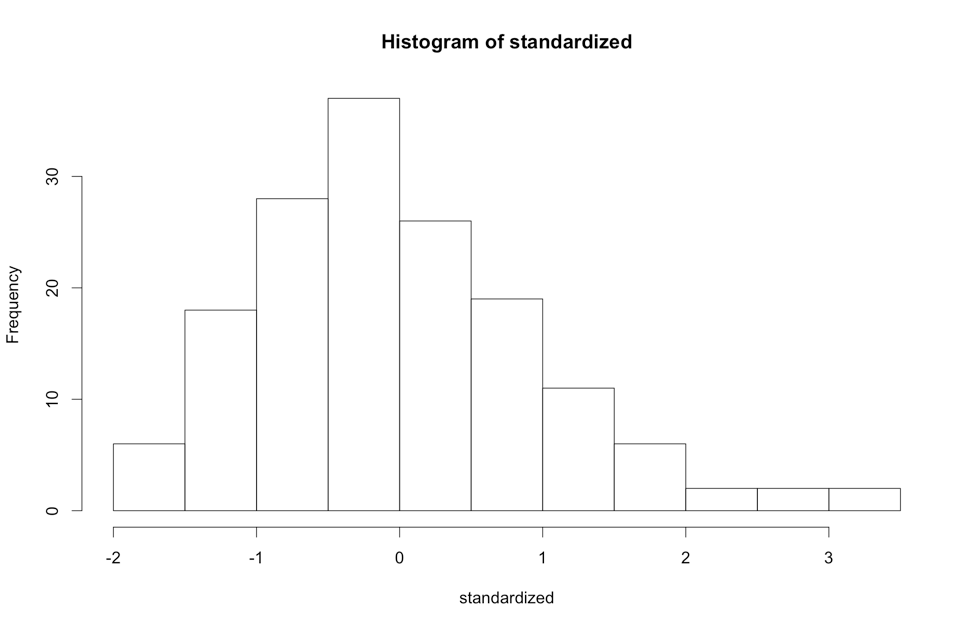


* 1. The ones marked in blue are ok, they are the column correlated with itself (which should be 1).
  2. So, our numbers appear ok.
  3. You can run correl to see the numbers, and use the LOWEST one for power if this is a pilot study.
     1. Here we would use .62 as the correlation between repeated measures, which means I overestimated for power.
     2. Check your learning – if you go back and estimate with .62, you should get 104 participants.

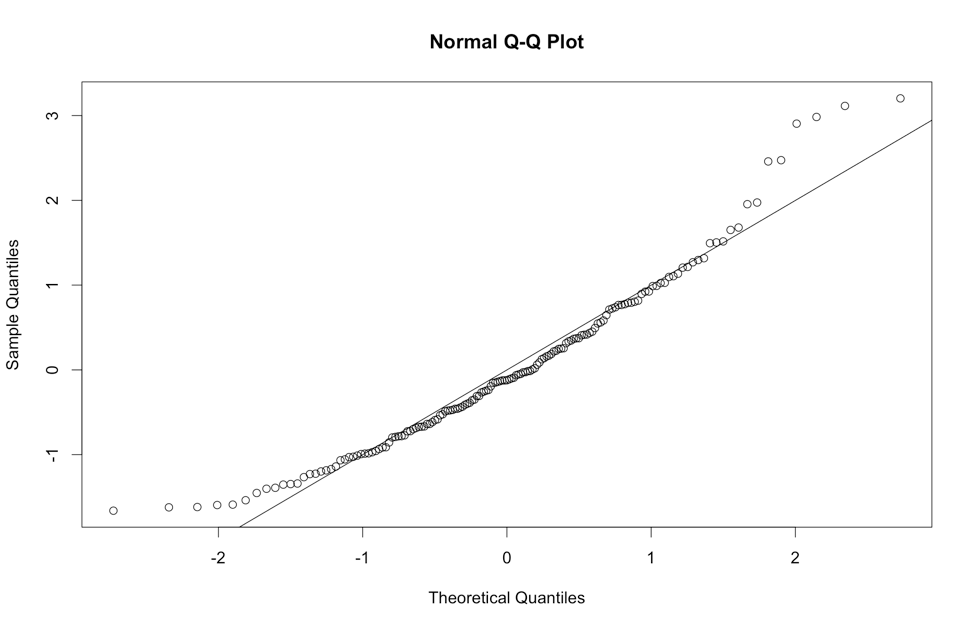




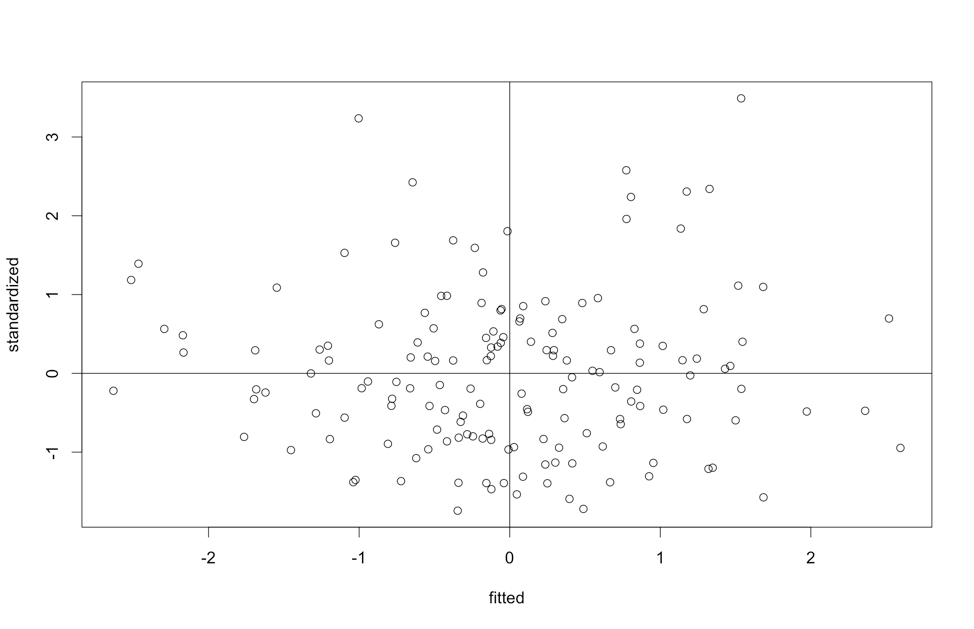
1. Set up the rest of the assumptions:
   1. Make a random variable:
      1. random = rchisq(nrow(*dataset*), 7)
   2. Run a fake regression:
      1. fake = lm(random~., data = *dataset*)
   3. Create the standardized residuals:
      1. standardized = rstudent(fake)
   4. Create the fitted values:
      1. fitted = scale(fake$fitted.values)
2. Normality:
   1. hist(standardized)
   2. Most of the data is between -2 and 2 and is centered over 0 – this chart is great!



* 1. Linearity:
     1. qqnorm(standardized)
     2. abline(0,1)
     3. This graph looks ok until we get to the tails of the distribution.
     4. I’d probably say this graph was alright.



* 1. Homogeneity:
     1. plot(fitted,standardized)
     2. abline(0,0)
     3. abline(v = 0)
     4. Here the data is not really homogeneous ... see how the top has a wider spread than the bottom (if you use vertical zero as the ruler). Horizontally, it’s ok.
     5. Now, most people do not talk about homoscedasticity for ANOVA, because homogeneity sort of equals homoscedasticity when one variable is categorical, and the other is continuous (aka the ANOVA set up).



* 1. Homogeneity: Take 2 Mauchly’s Test for Sphericity

1. Mauchly’s is a test for homogeneity between repeated measures (so to speak), which is called sphericity.
2. The assumption is considered *compound symmetry*:
   * + 1. The correlations between all the levels are equal.
       2. The variance of the difference scores between each level combination is the same.
     1. It is almost impossible to meet this assumption:
        1. Generally, you are examining if there are differences in levels.
        2. They are often taking the same thing over and over.
        3. So, the variances often get much smaller or larger across the levels.
        4. It’s such a problem, people often ignore sphericity.
     2. You will get this test automatically with the ANOVA output.
     3. NEW: Something important!
        1. IF there are only two levels of an IV, you will NOT get Mauchly’s test. Why? It’s hard to compare the correlation and variance of the difference scores if there are only two levels (because there is only one correlation and one difference score).

**Running the ANOVA:**

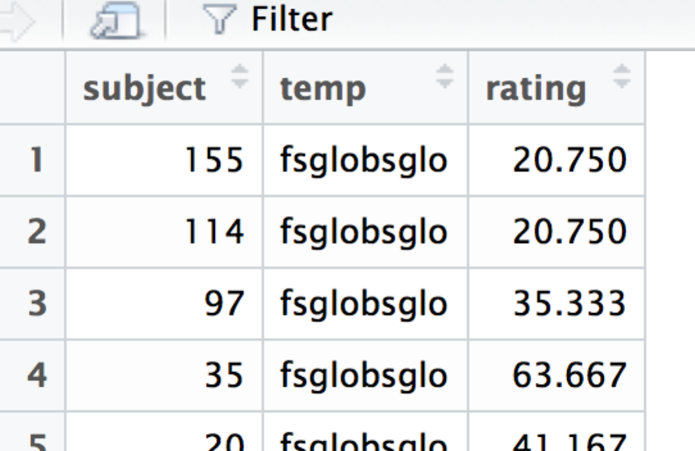
1. First, we must add a participant number to your data if it does not have one.
   1. We have one! Moving on.
2. Second, we must switch from WIDE to LONG format.
   1. Long format for repeated measures means that each level + participant get their own row … so that there is one column for the IV and one column for the DV.
   2. Install / load the reshape library (NOT reshape2).
   3. library(reshape).
   4. Melt the data (run all these lines):

longdata = melt(*dataset*,

id = "partno",

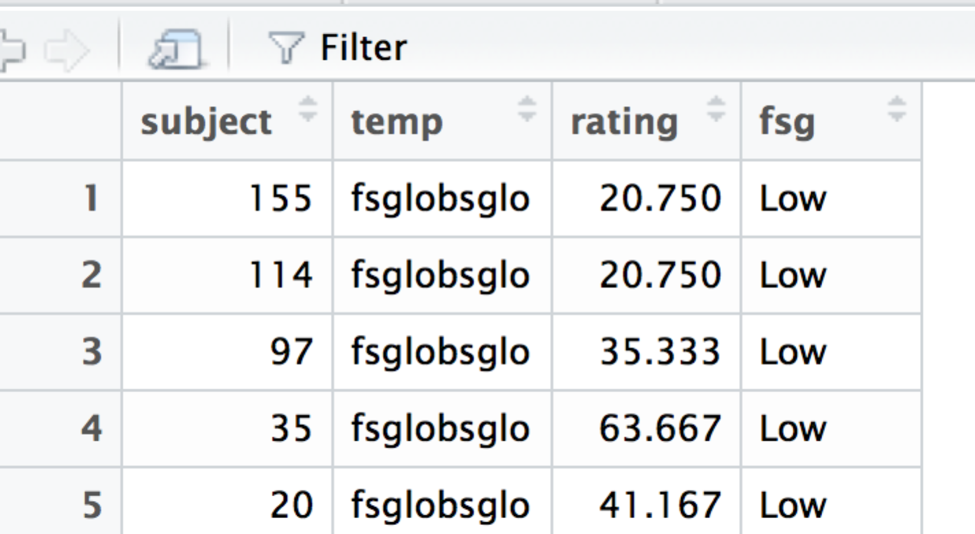
measured = c("*level column*", " *level column* ", " *level column*"))

* 1. Now, you should see that each column before is a new factored column (variable), and the DV is all one column (value).

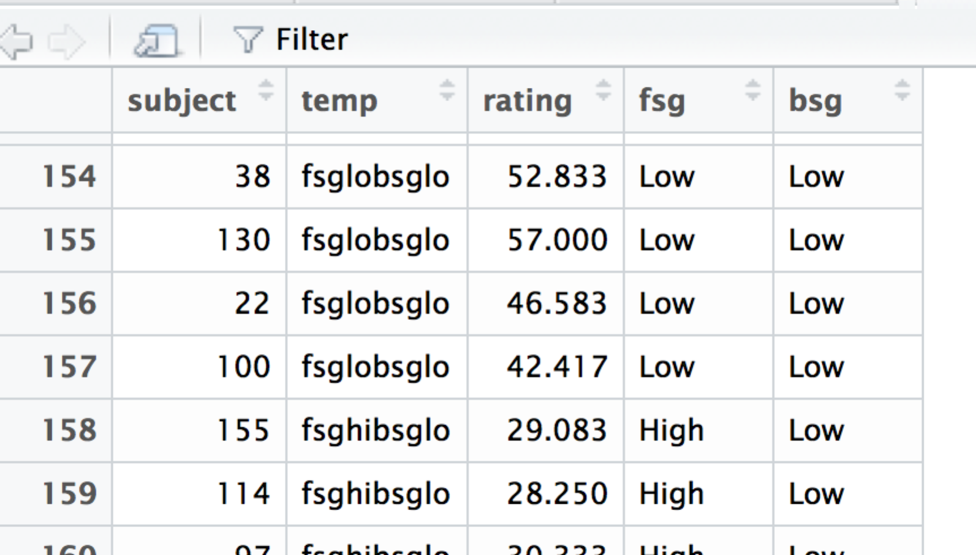


* 1. I’d suggest relabeling the column names since variable and value are not that helpful – change *column* out here to the new names.
     1. colnames(*dataset*) = c(“*column”, “column”, “column”*)

1. Create two new IV columns!
   1. Do not overwrite one of the columns you have now – you will use it to check what you are doing.
   2. Figure out the pattern of data:
      1. Look at how you melted the data, and how many rows you had originally.
      2. For FSG, I have lo hi lo hi … with N = 157.
         1. Therefore, I need to create a column that says LOW 157 times, then HIGH 157 times, then LOW 157 times, and HIGH 157 times.
         2. *dataset*$*columnname* = c(rep(*level*, *N*), rep(*level*, *N*), rep(*level*, *N*), etc.)
         3. The total number of repeated items and dataset must match or you will get an error.
         4. The rep function repeats the first argument as many times as the second argument.
            1. Therefore, rep(X, 5) = XXXXX.
   3. Check out the results:



* 1. Add the second variable. Scroll down to make sure that they match up correctly.
     1. I am comparing my temp column to the fsg and bsg columns to make sure it broke apart correctly.

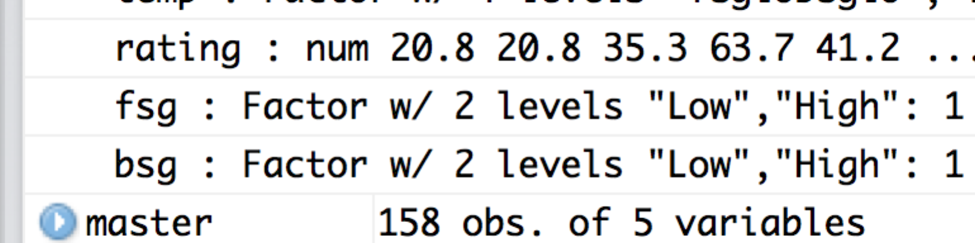


1. Factor those variables:
   1. Right now the variables we just added are listed as chr – characters. We will need them to be factors for some of the next steps to run correctly.
      1. *dataset$column* = factor(*dataset$column*,

levels = c(“*stuff”, “stuff”*),

labels = c(“stuff”, “stuff”))

* 1. Factoring also solves another problem we have – we want the LOW level to be first, then the HIGH level. However, R often interprets in alphabetical order, so we can reorder them using the factor command. Just put them in the order you want – making sure that levels and labels are in the same order.



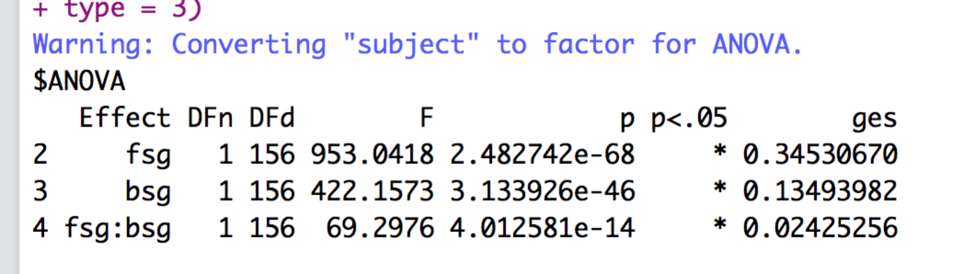
1. Load the ez library.
   1. library(ez)
2. Run the ANOVA (all these lines):
   1. ezANOVA(data = *dataset*,

wid = subject, ##participant number

within = .(*column of IV1, column of IV2),*

dv = *column of DV*,

type = 3)



1. Interpret the output:
   1. IF levels > 2, check Mauchly’s for Sphericity – You want p > .001.
      1. Look at the last number under *p*, which says .0015, that’s really close to being bad.
      2. If the *p* value in the Mauchly’s test is bad, you go on to look at the Sphericity Corrections right below it.
   2. Corrections:
      1. GGe = Greenhouse Geisser epsilon.
      2. p[GG] = p value if you used the GG corrections, with the \* to indicate p < .05.
      3. HFe = Huynh-Feldt epsilon.
      4. p[HF] = p value if you used HF correction, with the \* to indicate p < .05.
      5. If both the GG and Huynh-Feldt epsilons are < .75, then use GG.
      6. If >.75, then use Huynh-Feldt.
      7. You would report the ANOVA statistics, as described below, then say you corrected with Greenhouse-Geisser or Huynh-Feldt and list the corrected p value.
   3. Check the Omnibus (overall) test for your IVs:
      1. We have three of them! What happened?!
         1. You will get one *F* test for each IV and then also the interaction.
         2. You will interpret each one separately.
         3. Remember, if the interaction is significant, only do post hocs for the interaction.
      2. The DFn = df numerator or model.
      3. The DFd = df denominator or error.
      4. F = F
      5. p = p value.
      6. p < .05 helpfully tells you if it’s significant at *p* < .05, which is what we want to find.
      7. ges = generalized eta squared or *ηges2*.
      8. It’s significant yay!
      9. Write that up:
         1. FSG: *F*(1,156) = 953.04, *p* <.001, *ηges2* = .35.
         2. BSG: *F*(1,156) = 422.15, *p* <.001, *ηges2* = .13.
         3. Interaction: *F*(1,156) = 69.30, *p* <.001, *ηges2* = .02.
2. If the overall test is significant, you will have to run post hocs to figure out what happened – but remember the caveat at the beginning about what to do if the interaction is significant.
   1. First, I find it easiest to create a table to figure out my effects – and what is being compared. Remember that we are going to calculate each *pairwise* combination, which is every mean compared to every other mean.
   2. To get the means and SDs, we can use tapply.
      1. tapply(*dataset$DV*, list(*dataset*$*IV, dataset$IV*), mean)
      2. tapply(*dataset$DV*, list(*dataset*$*IV, dataset$IV*), sd)
      3. tapply(*dataset$DV*, list(*dataset*$*IV, dataset$IV*), length) – these should be all the same because everyone is in all groups, but it helps to see how many of each there were originally (rather then the 54 lines the dataset is now because we switched to long format).
      4. The first variable runs along the side, the other across the top.

MEANS

Low High

Low 50.23169 56.10342

High 64.30092 77.97475

SDs

Low High

Low 12.69984 12.56632

High 12.50383 11.85959

N

Low High

Low 157 157

High 157 157

* 1. Now, we have to figure out what to do about the interaction.
     1. Remember, if you are just looking at main effects, you can run tapply with only one of the IV variables to look at those means.
     2. We have 2X2 ANOVA – so we have four boxes to consider:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | BSG | |
|  |  | LOW | HIGH |
| FSG | LOW | LOW, LOW | LOW, HIGH |
| HIGH | HIGH, LOW | HIGH, HIGH |

* + 1. How do we want to compare them?
    2. The rule usually is across or down but not both.
    3. Once you pick a direction, you will need to SPLIT the dataset into chunks to analyze each piece separately.
       1. Use the subset function!
       2. Be sure to look at your N values to make sure they changed – it should total up to the full number of rows in longdata.

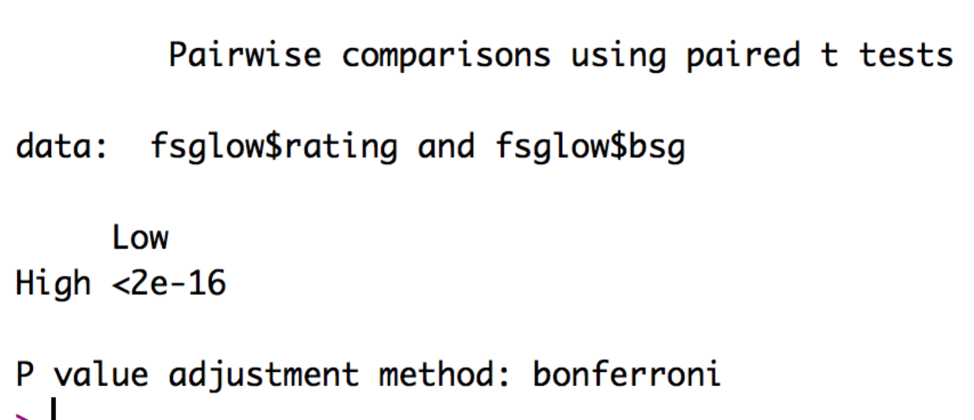
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mean 1 | Mean 2 | P-value | Explain? | Effect size |
| FSG LOW  BSG LOW  M = 50.23  SD = 12.70  N = 157 | FSG LOW  BSG HIGH  M = 56.10  SD = 12.57  N = 157 |  |  |  |
| FSG HIGH  BSG LOW  M = 60.30  SD = 12.50  N = 157 | FSG HIGH  BSG HIGH  M = 77.97  SD = 11.56  N = 157 |  |  |  |

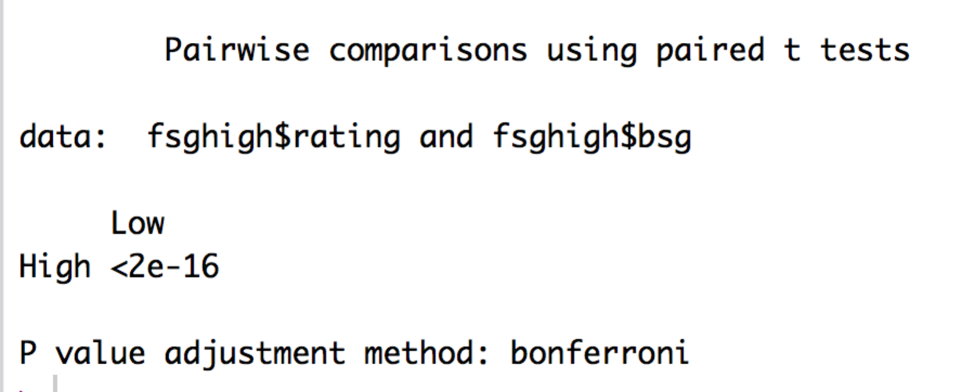
* 1. Now, we have to calculate the *post hoc test* and *post hoc correction* to find out what’s going on.
  2. Use the pairwise.t.test() function to run t.test you learned earlier on all groups at once.
     1. Remember, you use paired = T for **dependent** t-tests, which is what we want to use for **repeated-measures** ANOVA.
     2. p.adjust.method is the *correction*.
     3. pairwise.t.test(*dataset*$*DV*, *dataset*$*IV*,

paired = T,

p.adjust.method = "bonferroni")

* 1. Be sure to use the new dataset! You will have to do the same thing for each dataset that you just created.
  2. Check your learning here – make sure it says paired t-test – we need all the follow up tests to line up with the right type of ANOVA.
  3. Remember that Bonferroni changes the p values biased on the number of tests you are running. That’s good for us, because then we can use p<.05 again to determine if it is significant.
  4. Remember you can turn off scientific notation with:
     1. options(scipen = 999)
     2. You’ll see that those p-values are really small!
  5. Use the Bonferroni output to fill in your p-values.





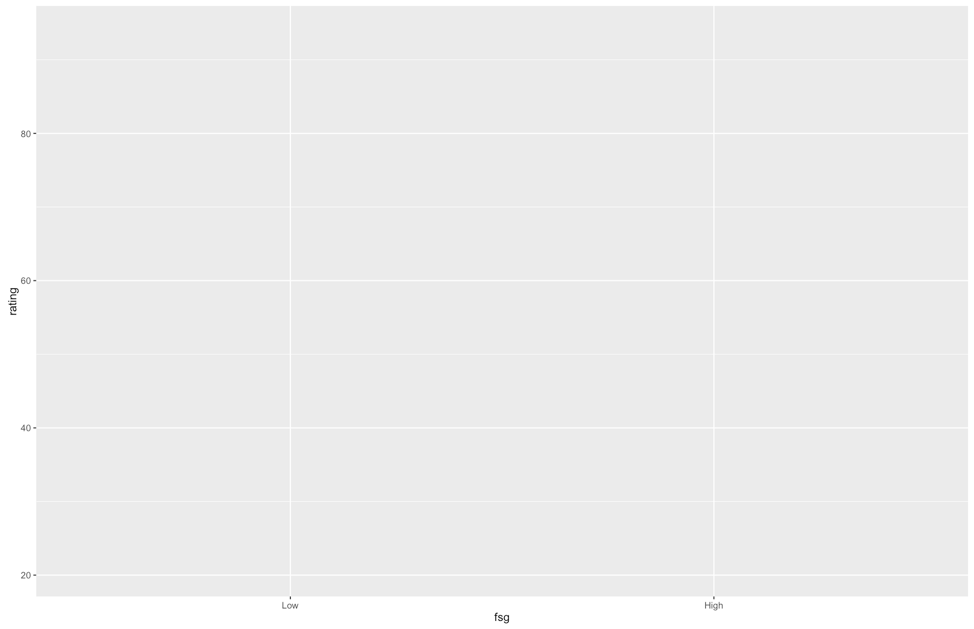
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mean 1 | Mean 2 | P-value | Explain? | Effect size |
| FSG LOW  BSG LOW  M = 50.23  SD = 12.70  N = 157 | FSG LOW  BSG HIGH  M = 56.10  SD = 12.57  N = 157 | <.001 | Significant,  High bsg > low bsg |  |
| FSG HIGH  BSG LOW  M = 60.30  SD = 12.50  N = 157 | FSG HIGH  BSG HIGH  M = 77.97  SD = 11.56  N = 157 | <.001 | Significant,  High bsg > low bsg |  |

* 1. You can use MOTE to calculate the effect sizes by loading the MOTE library.
     1. library(MOTE)
  2. Wait … you said this analysis was an interaction, which should have **different** patterns of data by levels. We got the same results that High > Low for BSG, even if FSG was low or high.
     1. Effect size to the rescue!
     2. We can talk about the differences in size of the effect for each comparison.
  3. We will use d.dep.t.avg for these calculations (yes, three times) because it uses the numbers we have (m, n, sd) for each time, rather than differences.
     1. d.dep.t.avg(m1 = #, m2 = #, sd1 = #, sd2 = #, n = #, a = .05)
     2. We have means and SDs in our table, and length should be the same across levels because they are the same people.
  4. Make sure each M, SD, and N look correct.
  5. Enter *d* only into your table.
  6. You can make *d* values positive or negative – I tend to report them as always positive because the negative just indicates that you subtracted the smaller mean first, not anything about the actual effect size.
  7. The effect size is TRIPLE for the high FSG pairs – that’s why the interaction is significant – while the pattern is the same (high > low), the strength of that pattern is different.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mean 1 | Mean 2 | P-value | Explain? | Effect size |
| FSG LOW  BSG LOW  M = 50.23  SD = 12.70  N = 157 | FSG LOW  BSG HIGH  M = 56.10  SD = 12.57  N = 157 | <.001 | Significant,  High bsg > low bsg | 0.46 |
| FSG HIGH  BSG LOW  M = 60.30  SD = 12.50  N = 157 | FSG HIGH  BSG HIGH  M = 77.97  SD = 11.56  N = 157 | <.001 | Significant,  High bsg > low bsg | 1.47 |

**Graphs:**

1. The best type of chart for anything analyzing group means is a bar chart with error bars.
2. We are going to use ggplot2 to build all our graphs.
   1. The package works like a transparency machine – you build layers and add them to the graph. You will really want to learn to stack your code, so that it’s easy to troubleshoot any problems you have.
3. First, load the ggplot2 library.
   1. library(ggplot2).
4. Create a blank graph with the right variables.
   1. X = IV, Y = DV.
   2. bargraph = ggplot(*datasetname,* aes(*Xcolumn, Ycolumn,* fill = *IVcolumn*))
   3. Note: fill has to be a factored variable. This variable will be put into a legend.
   4. Check that it worked – try running just bargraph. You should get a blank plot like this:



1. Which one should be the legend versus X axis?
   1. I put my split variable for interactions on the X axis, so the post hoc tests match the bars that are paired together.
2. Add things to the plot – note this code has changed a little bit:

bargraph +

stat\_summary(fun.y = mean,

geom = "bar",

position = "dodge") +

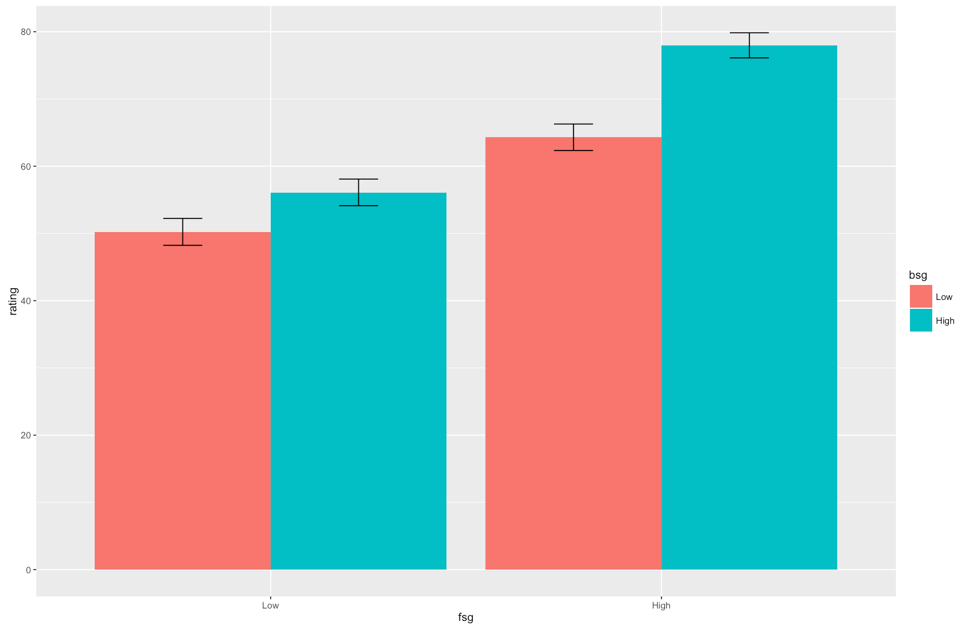
stat\_summary(fun.data = mean\_cl\_normal,

geom = "errorbar",

position = position\_dodge(width = 0.90),

width = 0.2)

* 1. Please note:
     1. That code above stays exactly the same, but remember that “” doesn’t copy correctly sometimes.
     2. What does it do?
        1. The first stat\_summary adds the bars to the graph by graphing the mean for each group.
        2. The second stat\_summary adds the error bars of the confidence interval (approximately 2\*SE). These bars help you see how much the variance is spread around each group.
     3. You should have this now:



* 1. That is the right graph, but it is **hideous.**
  2. First, we are going to clean up the gray background, the nondiscriminate axes, and the tiny type.
  3. Separate from the graph code, run this code exactly:

cleanup = theme(panel.grid.major = element\_blank(),

panel.grid.minor = element\_blank(),

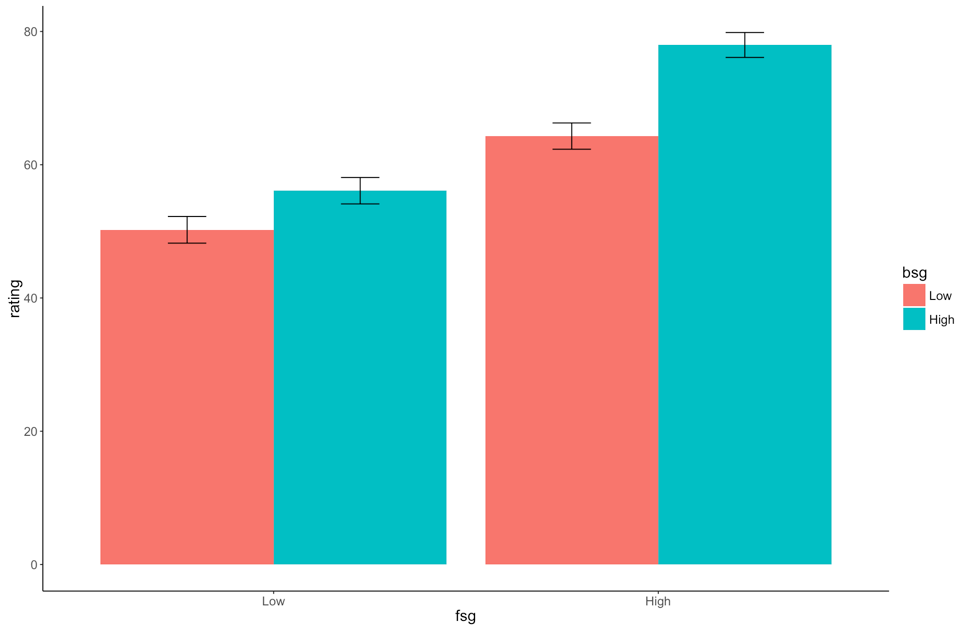
panel.background = element\_blank(),

axis.line = element\_line(colour = "black"),

legend.key = element\_rect(fill = "white"),

text = element\_text(size = 15))

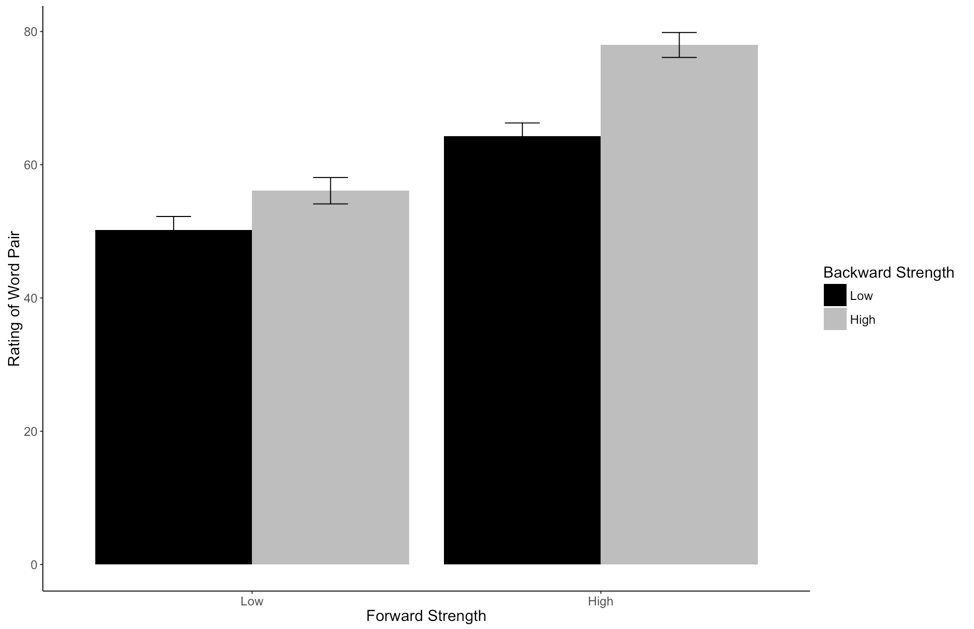
* 1. This code saves a whole bunch of settings as theme, which then we can add to our graph.
  2. NOTE: In this demo, we are walking through one part at a time, but you will run the entire graph code again to recreate the graph. It isn’t quite cool enough to remember what you did a minute ago.



* 1. Ok, that’s better, but now we have two issues:
     1. The x and y axes labels are terrible – what do they even mean?
  2. How to fix that:
     1. xlab(“Text that you want”) + ylab(“Text that you want”) will fix the axes labels.
  3. Next issue – the bad looking bsg on the legend and colors:
     1. Notice in the first line we created the graph, we used the word FILL.
     2. We can do scale\_fill\_manual to fix that problem. The name part will change the overall label, and you can use labels if you want to fix the level labels.
     3. You can also make it black / gray / white / green / purple by using the values command.
     4. scale\_fill\_manual(name = c(“*Name of IV*”),

labels = c(“*level”, “level” ,…*),

values = c(“*color*”, *“color”, …*))

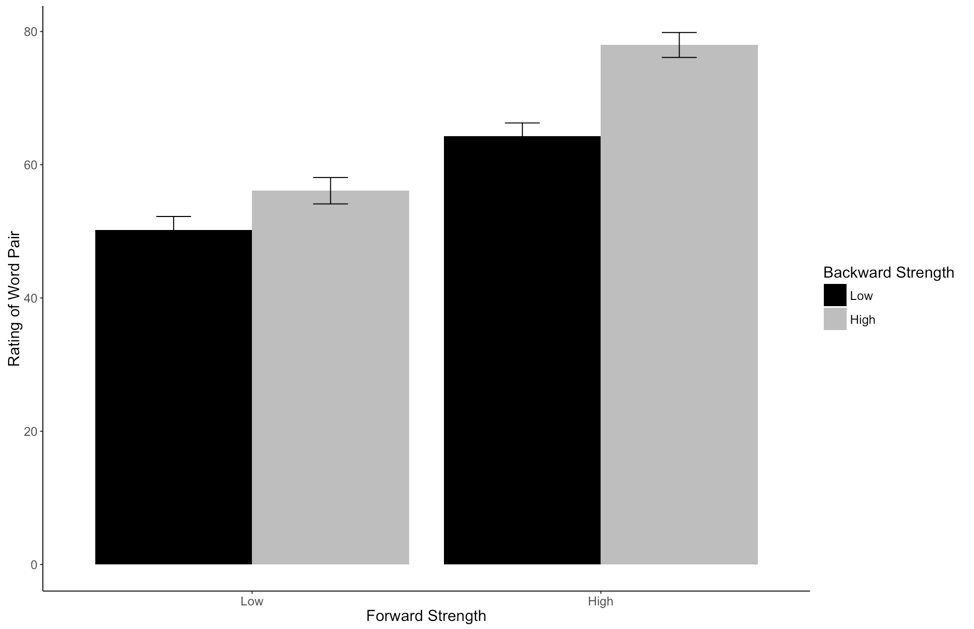


**Example write up:**

**Results**

Participants were given pairs of words and asked to rate them on how often they thought 100 people would give the second word if shown the first word. The strength of the word pairs was manipulated through the actual rating (forward strength: FSG) and the strength of the reverse rating (backward strength: BSG). Data was screened for assumptions and outliers. One multivariate outlier was found using Mahalanobis distance as a criterion, and was excluded in the analysis.

Figure one shows the mean ratings for each combination of low and high forward and backward strength. FSG condition had a significant effect on ratings, *F*(1,156) = 953.04, *p* <.001, *ηges2* = .35, which indicated that participants could correctly rate low related pairs (*M* = 53.17, *SD* = 12.95) lower than high related pairs (*M* = 71.14, *SD* = 13.96). However, BSG condition had a significant effect on ratings, *F*(1,156) = 422.15, *p* <.001, *ηges2* = .13. Participants again were able to rate low related pairs (*M* = 57.27, *SD* = 14.42) lower than high related pairs (*M* = 67.04, *SD* = 16.39). Finally, the interaction between FSG and BSG was significant, *F*(1,156) = 69.30, *p* <.001, *ηges2* = .02. Dependent t-tests were used for the post hoc analysis of the interaction between FSG and BSG with a Bonferroni correction. BSG had an approximately six point effect on ratings for low FSG pairs, *p*<.001, *davg =* 0.46. However, for high related FSG pairs, BSG increase from low to high also increased ratings about 18 points, *p*<.001, *davg =* 1.47. See Figure 1 for the interaction.



*Figure 1.* The interaction between forward and backward strength in estimating the relationship between word pairs.