# Complete Example (Repeated Measures – 1 IV only)

A note on repeated measures designs:

* People sometimes find it difficult to separate IV and DV in these designs because they are related. Always make sure you can differentiate between the levels of the IV and what was measured at each of those levels (i.e. the DV).
* The other issue is that data is often given to you in WIDE format, rather than LONG format.
  + You will want to data screen in WIDE format.
  + You will have to run the ANOVA and graphs in LONG format.
* These designs are more powerful (need less people!) because you are measuring the same people a couple times. That means that you reduce the error variance (within subjects) because you can control for the fact that everyone is slightly different and take that out of the error variance. However, the draw back is that there is an interaction with the study sometimes (i.e. if you take the same test over and over, of course you’ll get better … or worse because you are bored.). So, you have to balance these effects (carry over effects, fatigue) with the ability to control for participant wackiness.

Participants were tested over several days to measure variations in their pulse given different types of stimuli. One stimulus was a neutral picture (like a toaster), while other stimuli were cute/happy pictures (puppies, babies), and negative stimuli (mutilated faces, pictures of war). Where there differences in pulse for each participant across the stimuli?

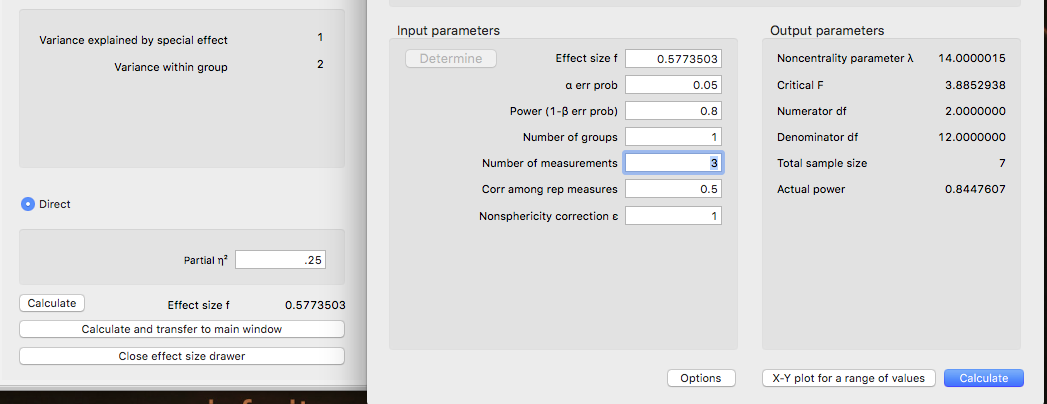
**Datafile:** rm 1 anova.csv

**IV:** Stimuli type – neutral, negative, positive

**DV:** Pulse/heart rate

**Power:**

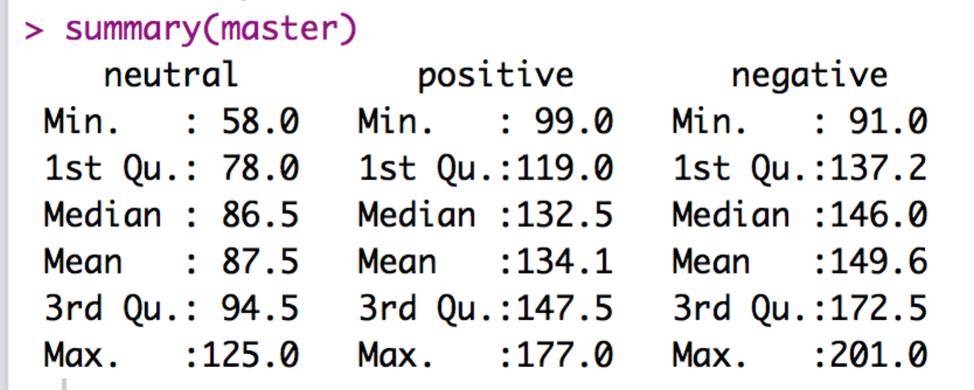
1. Open Gpower!
   1. Test family: F-test
   2. Statistical Test: ANOVA repeated measures, within factors
   3. Estimate an effect size: click determine 🡪 click direct 🡪 use eta square sizes you think might be accurate, remember small, medium, and large estimates from the notes.
   4. Alpha = .05
   5. Power (1-beta .20) = .80
   6. Number of groups = number of IVs
   7. Number of measurements = number of levels
   8. Corr among rep measures = correlation between levels
      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 … 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. Large effect size
   2. One IV
   3. Levels from our current study
   4. Correlation = .5
   5. Epsilon = 1
3. Says we needed to run 7 people to find a significant effect with a large effect size!

****

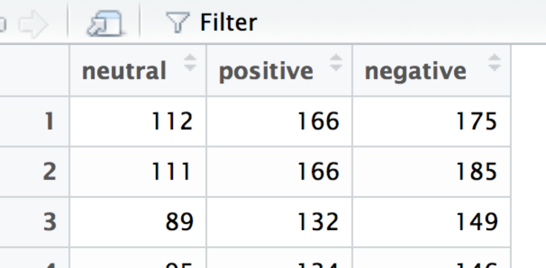
1. Check your learning:
   1. See if you can estimate a small effect size with a high correlation between measures (*r* = .80). You should get that you need 66 people.

**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. No one appears dead (too slow), but some of them do appear sort of high (I think you might explode). We are going to assume they are accurate (because this data is made up data).



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. 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.

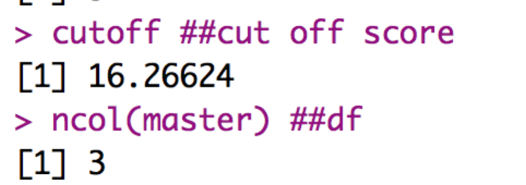


* 1. Because they are in this format, we have several columns to work with, which means we can use Mahalanobis values. We want to use this format for data screening because it accounts for the fact that people have more than one measurement. We would not want to ignore that person one is person one for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

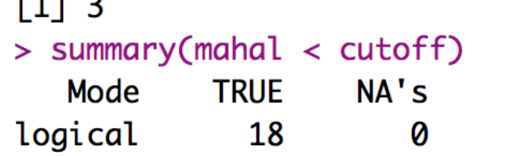
colMeans(*dataset*, na.rm = T),

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*

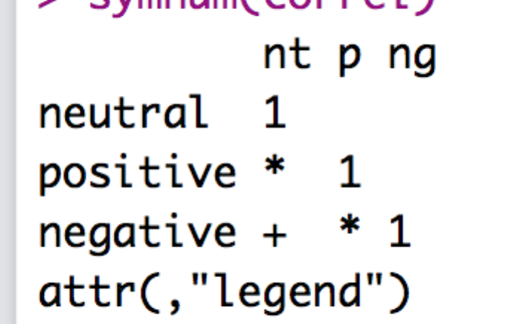


* 1. See how many outliers you have:
     1. summary(mahal < cutoff)

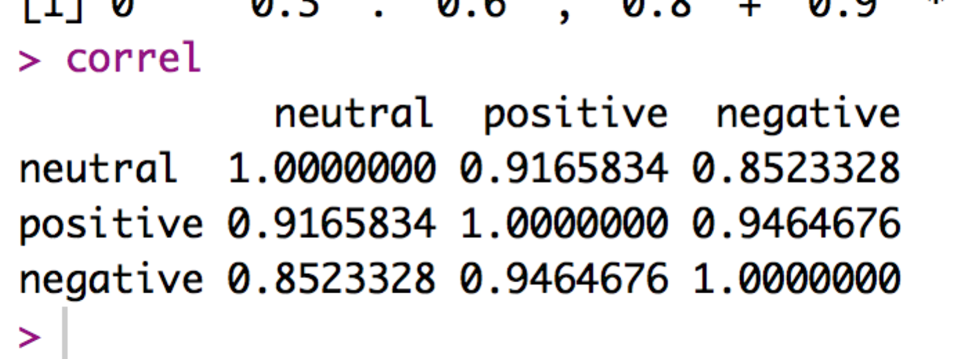


* + 1. Remember FALSE is bad.
  1. Exclude outliers:
     1. noout = subset(*dataset*, mahal < cutoff)

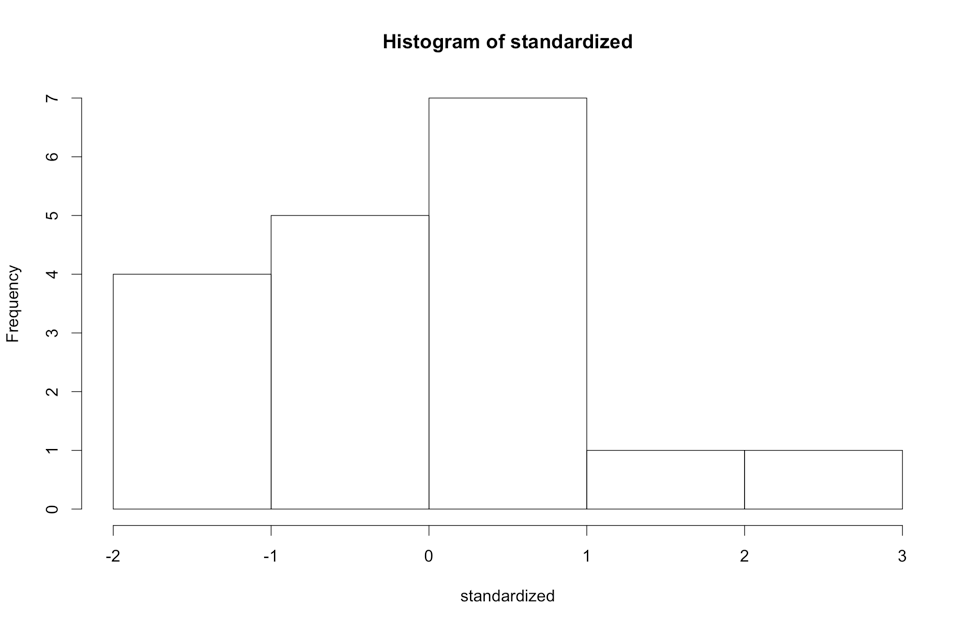
1. Additivity
   1. We do have to worry about correlations in a repeated measures design, but not quite in the same way we talked about it for overall data screening.
   2. In general, you *want* the various measurements to be highly correlated – it will give you more power if they are correlated and 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:



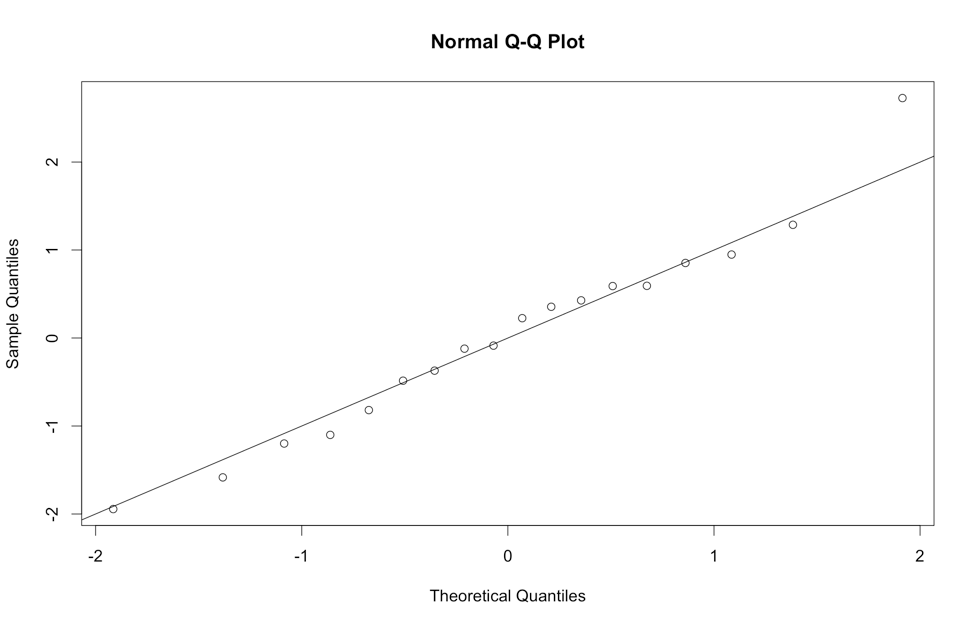
* 1. The ones marked in blue are ok, they are the column correlated with itself (which should be 1).
  2. So, our numbers appear ok.
  3. You can run correl to see the numbers, and use the LOWEST one for power if this is a pilot study.
     1. Here we would use .85 as the correlation between repeated measures.



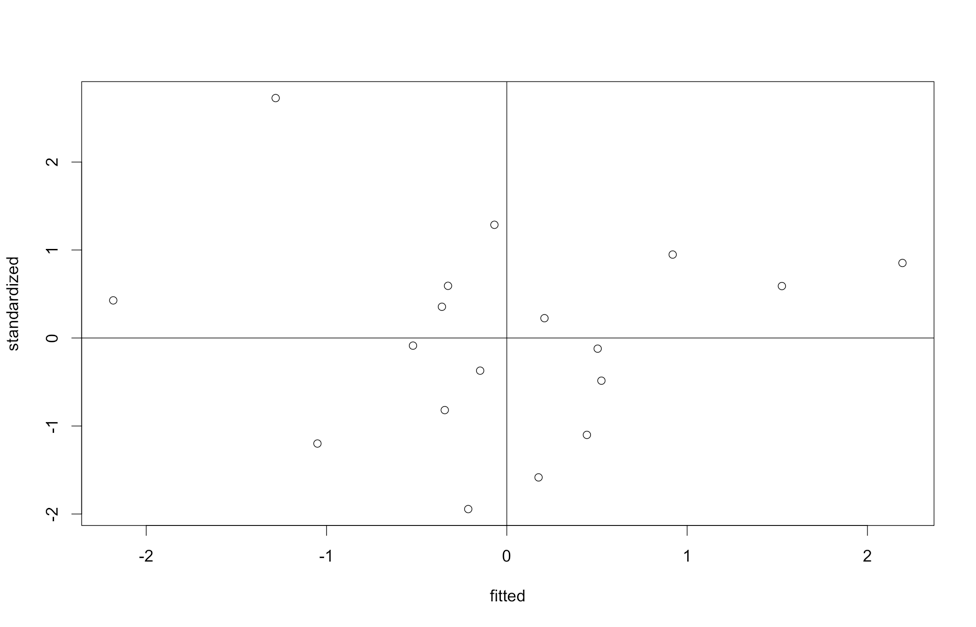
1. Set up the rest of the assumptions:
   1. Make a random variable:
      1. random = rchisq(nrow(*dataset*), 7)
   2. Run a fake regression:
      1. fake = lm(random~., data = *dataset*)
   3. Create the standardized residuals:
      1. standardized = rstudent(fake)
   4. Create the fitted values:
      1. fitted = scale(fake$fitted.values)
2. Normality:
   1. hist(standardized)
   2. It’s a bit positively skewed, but not too bad. It would be better if we had at least 30 people!



* 1. Linearity:
     1. qqnorm(standardized)
     2. abline(0,1)
     3. This graph looks pretty good.



* 1. Homogeneity:
     1. plot(fitted,standardized)
     2. abline(0,0)
     3. abline(v = 0)
     4. Here the data is ok, most of the holes are because of small sample size.
     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.

**Running the ANOVA:**

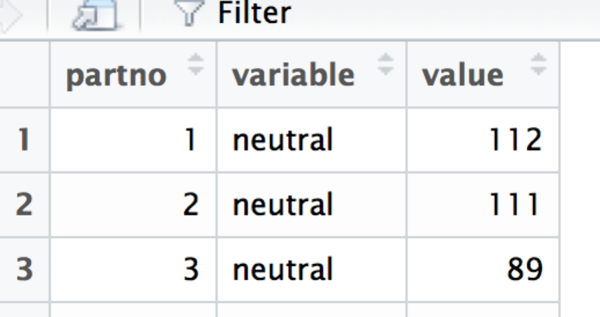
1. First, we must add a participant number to your data if it does not have one.
   1. The ez package requires a participant number, so we will have to add one.
   2. *dataset$partno* = 1:nrow(*dataset*)
   3. Yes, do this BEFORE the next step or your data will run incorrectly for the ANOVA.
2. Second, 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 (like partno, pictype, heartrate).
     1. colnames(*dataset*) = c(“*column”, “column”, “column”*)

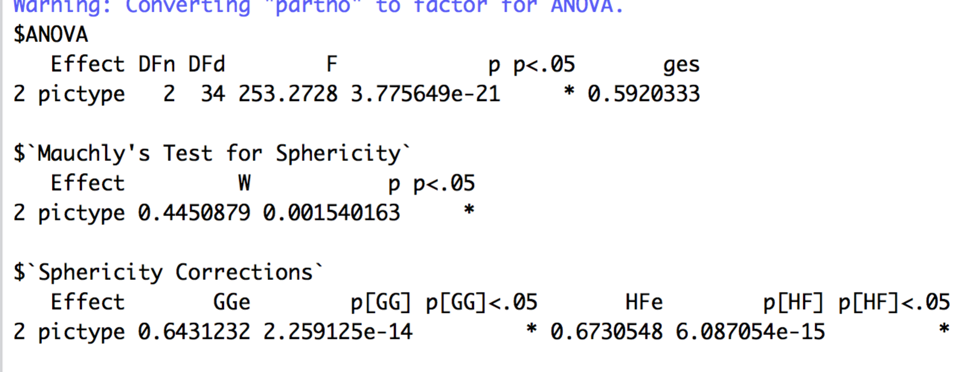
1. Load the ez library.
   1. library(ez)
2. Run the ANOVA (all these lines):
   1. ezANOVA(data = *dataset*,

wid = partno,

within = *column of IV,*

dv = *column of DV*,

type = 3)



1. Interpret the output:
   1. 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. Let’s talk about what you do if it’s bad, so you know how to correct for sphericity problems.
      3. 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 correction, 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 IV:
      1. Under effect it says *pictype* – that’s because my IV is named group. Therefore, it will say whatever the IV name is.
      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 square or η2.
      8. It’s significant yay!
      9. Write that up:
         1. APA: *F*(2,34) = 253.27, *p* <.001, η2 = .59, using a Greenhouse-Geisser correction for sphericity on *p*.
         2. AMA: *F*2,34 = 253.27, *p* <.001, η2 = .59, using a Greenhouse-Geisser correction for sphericity on *p*.
2. If the overall test is significant, you will have to run post hocs to figure out what happened.
   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*), mean)
      2. tapply(*dataset$DV*, list(*dataset*$*IV*), sd)
      3. tapply(*dataset$DV*, list(*dataset*$*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).

Means

neutral positive negative

87.5000 134.1111 149.5556

SD

neutral positive negative

16.73056 21.74961 27.74934

N

neutral positive negative

18 18 18

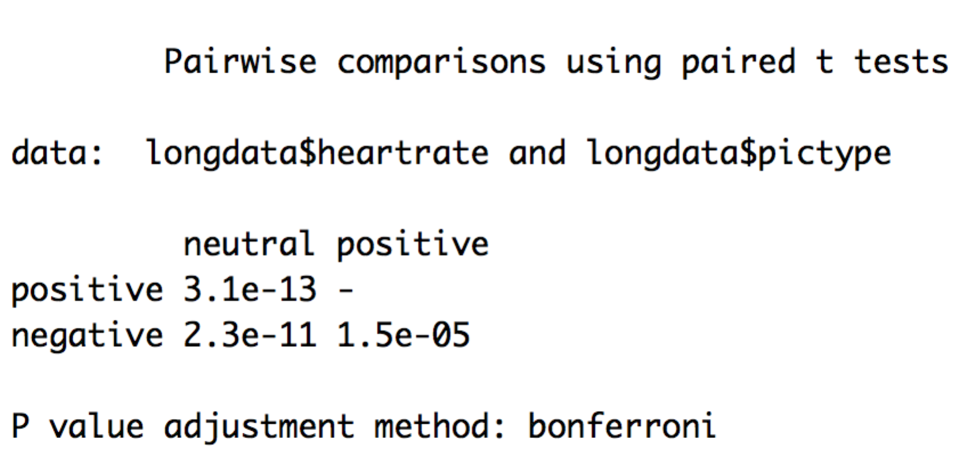
* 1. Let’s put that into a table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mean 1 | Mean 2 | P-value | Explain? | Effect size |
| Neutral  M = 87.50  SD = 16.73 | Positive  M = 134.11  SD = 21.74 |  |  |  |
| Neutral  M = 87.50  SD = 16.73 | Negative  M = 149.56  SD = 27.75 |  |  |  |
| Positive  M = 134.11  SD = 21.74 | Negative  M = 149.56  SD = 27.75 |  |  |  |

* 1. Now, we have to calculate the *post hoc test* and *post hoc correction* to find out what’s going on. Where are the differences in our heart rate? It seems that they should all be different just looking at the means, but we will have to test it!
  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. 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.
  2. 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.
  3. Remember you can turn off scientific notation with:
     1. options(scipen = 999)
     2. You’ll see that those p-values are really small!
  4. Use the Bonferroni output to fill in your p-values.

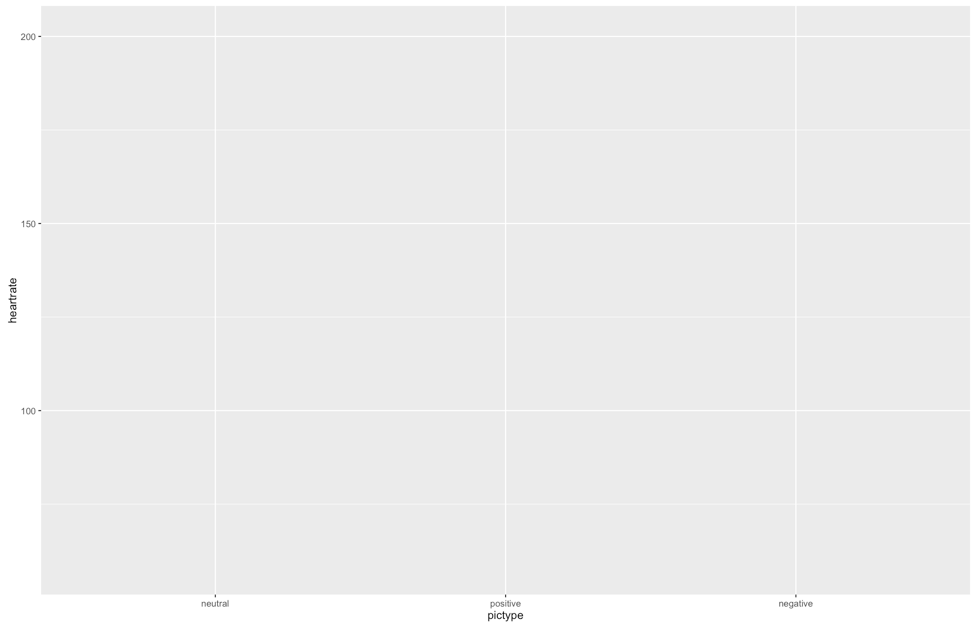
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mean 1 | Mean 2 | P-value | Explain? | Effect size |
| Neutral  M = 87.50  SD = 16.73 | Positive  M = 134.11  SD = 21.74 | <.001 | Significant,  Positive > heart rate than Neutral |  |
| Neutral  M = 87.50  SD = 16.73 | Negative  M = 149.56  SD = 27.75 | <.001 | Significant,  Negative > heart rate than Neutral |  |
| Positive  M = 134.11  SD = 21.74 | Negative  M = 149.56  SD = 27.75 | <.001 | Significant,  Negative > heart rate than Positive |  |

* 1. You can use MOTE to calculate the effect sizes by loading the MOTE library.
     1. library(MOTE)
  2. 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.
  3. Make sure each M, SD, and N look correct.
  4. Enter *d* only into your table.
  5. 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.
  6. It’s interesting to see here that pictures caused a BIG effect versus neutral, but only a medium effect against each other.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mean 1 | Mean 2 | P-value | Explain? | Effect size |
| Neutral  M = 87.50  SD = 16.73 | Positive  M = 134.11  SD = 21.74 | <.001 | Significant,  Positive > heart rate than Neutral | 2.42 |
| Neutral  M = 87.50  SD = 16.73 | Negative  M = 149.56  SD = 27.75 | <.001 | Significant,  Negative > heart rate than Neutral | 2.79 |
| Positive  M = 134.11  SD = 21.74 | Negative  M = 149.56  SD = 27.75 | <.001 | Significant,  Negative > heart rate than Positive | 0.62 |

**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*))
   3. Check that it worked – try running just bargraph. You should get a blank plot like this:



1. Add things to the plot:

bargraph +

stat\_summary(fun.y = mean,

geom = "bar",

fill = "White",

color = "Black") +

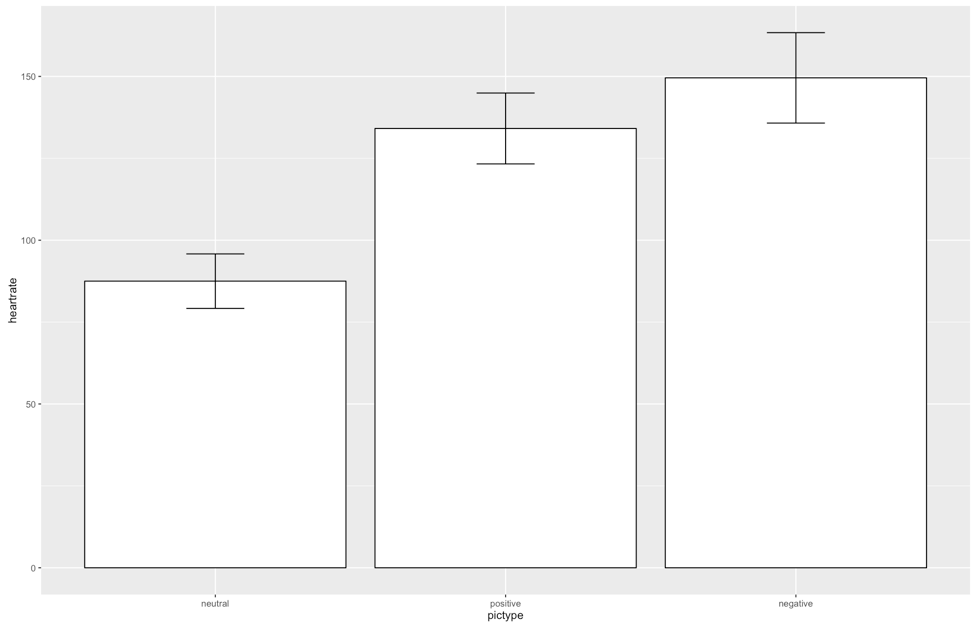
stat\_summary(fun.data = mean\_cl\_normal,

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.

bargraph +

stat\_summary(fun.y = mean,

geom = "bar",

fill = "White",

color = "Black") +

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

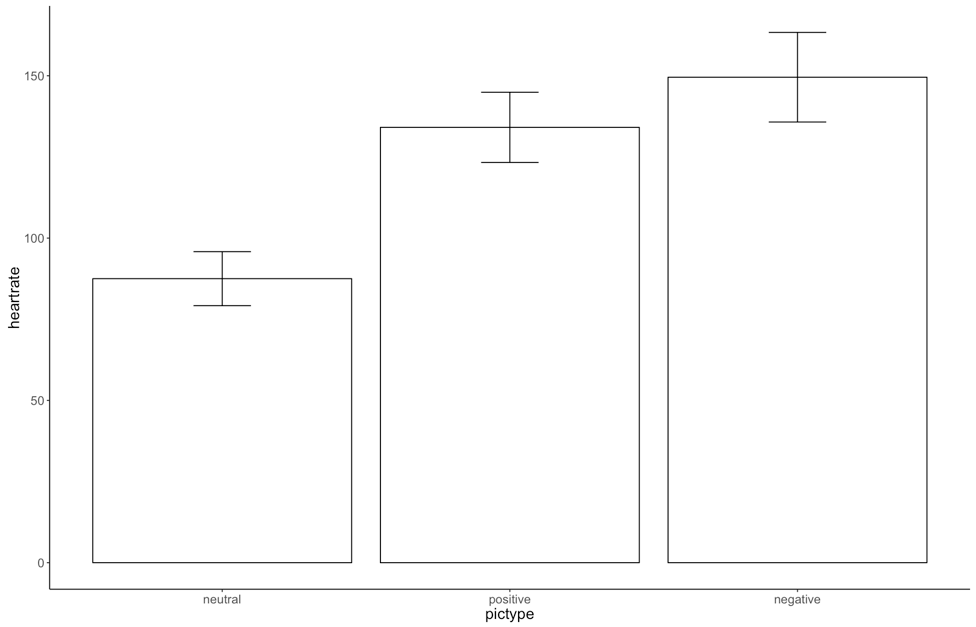
geom = "errorbar",

position = position\_dodge(width = 0.90),

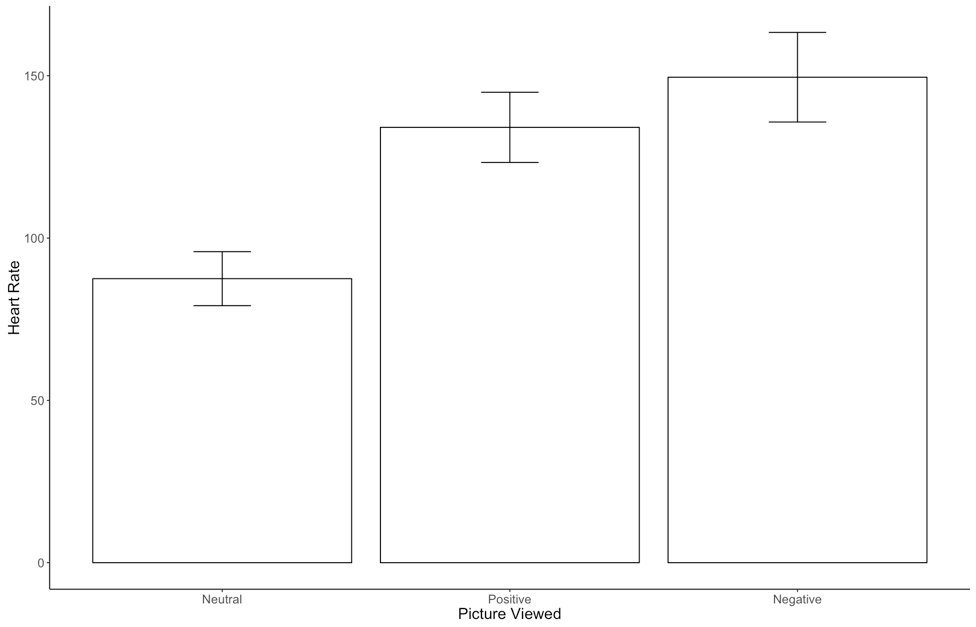
width = 0.2) +

cleanup

Should give you:



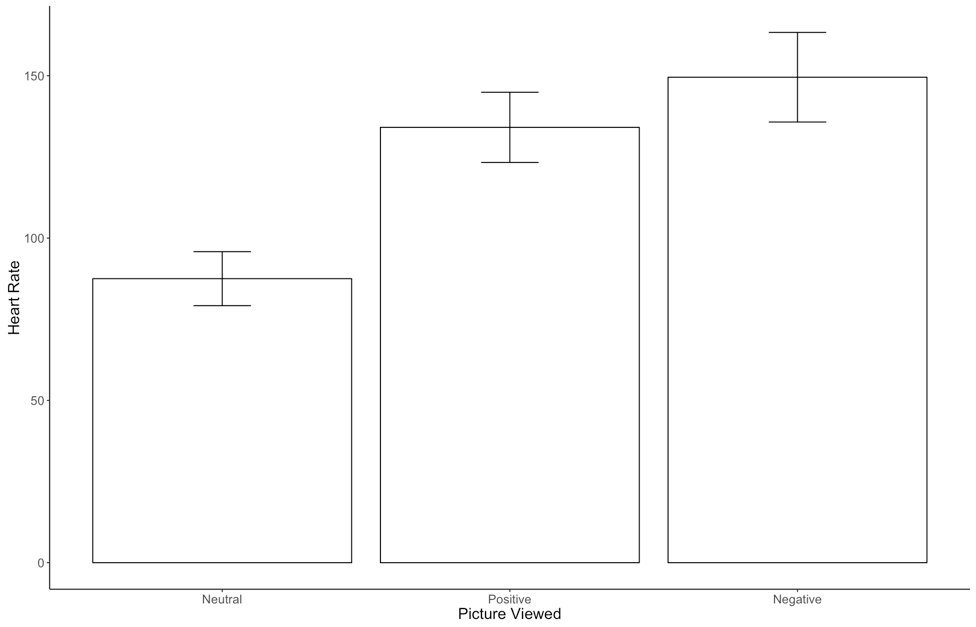
* 1. Ok, that’s better, but now we have two issues:
     1. The x and y axes labels are terrible – what do they even mean?
     2. The group labels are not capitalized.
  2. How to fix that:
     1. xlab(“Text that you want”) + ylab(“Text that you want”) will fix the axes labels.
     2. scale\_x\_discrete(labels = c(“Label1”, “Label2”, …)) will add labels – be sure to type them in the same order you have your bars currently – it does not rearrange, just relabels.
  3. If you run all the code from before and that code, you should get:



Example write up (note: this example does not have M and SD like the last one, because I have the figure, both ways are acceptable.).

**Results**

Participants were tested on three types of stimuli (neutral, positive, negative) for pulse rate. Data was screened for errors, missing data (none), outliers (none found with Mahalanobis distance), and assumptions. Normality, linearity, homogeneity, and Mauchly’s test (*p* = .002) indicated that assumptions were met. Using a repeated measures ANOVA, different stimuli were found have different heart rates, *F*(2,34) = 253.27, *p* <.001, η2 = .59. Post hoc comparisons were analyzed using dependent t-tests with a Bonferroni correction. Neutral stimuli were found to have a significantly lower mean than Positive stimuli (*p*<.001, *davg* = 2.42), as well as Negative stimuli (*p*<.001, *davg* = 2.79). Positive stimuli showed lower heart rates than Negative stimuli (*p*<.001, *davg* = 0.62). Therefore, heart rates were lowest at Neutral stimuli, followed by Positive stimuli, and highest with Negative Stimuli. Figure 1 displays the average heart rates along with confidence intervals.



*Figure 1.* Average heart rates with 95% confidence intervals for each stimuli type.