Post Hoc Analysis for One-way ANOVA

Neil J. Hatfield

3/19/2022

In this tutorial, we are going to explore doing Post Hoc analysis in the context of One-way ANOVA models. We will look at both Parametric and Nonparametric shortcuts. The general structure for this guide is as follows:

- Setting up `R` and Loading Data
  - Loading Packages, Setting Options, and Additional Tools
  - Loading Data
- Key Decisions for Post Hoc Analysis
  - Study Design Decisions
  - Checking Appropriateness and Assumptions
- Post Hoc for the Parametric Shortcut
  - Tukey's [& Kramer's] Honest Significant Difference
  - Pairwise Methods
  - `DescTools` Methods
  - Special Comparisons
  - Effect Sizes
- Post Hoc for the Nonparametic Shortcut
  - Dunn's Test
  - DSCF Test
  - Effect Sizes
- Visualizing Pairwise Differences

As a quick vocabulary note, we often use the term "omnibus" to refer to the hypothesis testing using the ANOVA F test or the Kruskal-Wallis H test. These tests are "omnibus" tests in that the alternative hypothesis is a family that contains several distinct hypotheses. Post Hoc analysis focuses on finding which of these hypotheses actually appears to fit our data.

Setting up `R` and Loading Data

As is always the case, we must do a bit of housekeeping to ensure that `R` will work for us nicely.

Loading Packages, Setting Options, and Additional Tools

For this tutorial/guide, we will make use of the following packages: `tidyverse`, `knitr`, `kableExtra`, `parameters`, `DescTools`, `dunn.test`, and `multcompView`.

We will also need to set our two options: telling `R` that table cells with missing values should visually look empty and that we require treatment/group effects for each factor to sum to zero.

Finally, we will want to load the set of tools I've created as there are two functions that will directly help us with Post Hoc analysis.

Here is a reminder for how to write this code in `R`:

```r
# Demo code for setting up R
# Load packages
packages <- c("tidyverse", "knitr", "kableExtra",
              "parameters", "DescTools", "dunn.test",
              "PMCMRplus", "multcompView")
lapply(packages, library, character.only = TRUE)

# Set options
options(knitr.kable.NA = "")
options(contrasts = c("contr.sum", "contr.poly"))

# Load additional tools
source("https://raw.github.com/neilhatfield/STAT461/master/rScripts/ANOVATools.R")
```

Loading Data

For this guide/tutorial, we are going to make use of two data sets: our Song Knowledge study and a new data set, **Free Amino Acids in Cheese** (or the Cheese study). For information on the Cheese study, look at Table 5.2 and Example 5.5 from page 91 of Oehlert.

Take a moment to think about how you would read in the data then compare your code with the following:

```r
# Demo code for loading data
# Song Knowledge study
songData <- read.csv(
  file = "https://raw.github.com/neilhatfield/STAT461/master/dataFiles/songKnowledge2022.csv",
  header = TRUE,
  sep = ","
)
# Set year to an ordered factor
songData$year <- factor(
  x = songData$year,
  levels = c("sophomore", "junior", "senior")
)

# Cheese study
# Create the Cheese data frame
cheeseData <- data.frame(
  strain = as.factor(
    c("None", "None", "A", "A",
      "B", "B", "(A & B)", "(A & B)")
  ),
  acids = c(
    4.195, 4.175, 4.125, 4.735,
    4.865, 5.745, 6.155, 6.488
  )
)

# Notice that I used as.factor in the data.frame call;
# this is another approach you can use
```

Key Decisions for Post Hoc Analysis

When we think about Post Hoc Analysis (either the pairwise explorations of this guide/tutorial or contrasts/linear combinations of a separate guide/tutorial), there are some steps that we need to take first.

Study Design Decisions

Our first steps for Post Hoc Analysis actually relate to our study design. Post Hoc analyses are different from our initial approach in that we engage in these analyses ***after*** we've seen the data. In fact, we only engage in Post Hoc analyses when we reject the null hypothesis for the (One-way) ANOVA model. To help guard ourselves against any claims that we might be up to problematic behaviors, we often add a couple of sentences to our study design such as the following:

> After getting and cleaning the data, we'll use the parametric ANOVA F test to answer the research question at a 5% level of significance. For Post Hoc analysis, we'll use an overall Type I Error Rate of 5% using Tukey's HSD.

Some of the above statement should feel familiar: 1) choice of the parametric shortcut, 2) statement of our Unusualness Threshold (level of significance), and 3) a statement of our overall Type I Risk. However, this third element has some slightly different language–"Error Rate" vs. "Risk". There is an additional element of method ("Tukey's HSD").

There are several different ways in which we can operationalize or quantify Type I Risk. We refer to these ways as "Type I Error Rates" and there are five common types. We have to pick one of these types of error rates to then select a method.

Testing Families

To help us pick a method, we also need to conceptualize our post hoc "testing family"; that is, if we reject the null hypothesis from the "omnibus" ANOVA test (i.e., F test or Kruskal-Wallis H), what combinations of groups/treatments do we want to test? Here are few examples of testing families:

- Pairwise Combinations
  - All possible pairs
    - Ex: Sophomores vs. Juniors, Sophomores vs. Seniors, Juniors vs. Sophomores
  - Special Pairs
    - Compare to Null Treatment
      - Ex: A vs. No starting strain, B vs. No starting Strain, (A & B) vs. No starting strain
    - Comparing to Standard Care
      - Ex: New Drug 1 vs. Current, New Drug 2 vs. Current, New Drug 3 vs. Current
    - Comparing to "Best" Treatment
- (Linear) Combinations ("Contrasts")
  - Ex: Graduating vs. Not Yet Graduating
    - I.e., Seniors vs. (Sophomores and Juniors)
  - Ex: Oral vs. Topical Pain Relief Medications
    - I.e., (aspirin, acetaminophen, ibuprofen) vs. (lidocaine, capsaicin, benzocaine)

In this tutorial/guide, we'll focus on pairwise families. (We'll have a separate tutorial/guide for the combinations.) One of the key aspects of conceptualizing your post hoc testing family is to count the number of comparisons, m, that make up the family.

Choose Your Type I Error Rate and Method

Once you conceptualize your testing family, you can yourself "Which Type I Error Rate do I want to control?" The following table shows the five different Type I Error Rates we can choose from. I've ordered the table from the most conservative to the least conservative; top to bottom. This order also reflects moving from the strongest guards against Type I Errors (top) to the weakest guards against Type I Errors (bottom).

Selected Type I Error Rate	Possible Parametric Methods
Simultaneous Confidence Intervals	Bonferroni, Tukey HSD, Tukey-Kramer HSD, Scheffé
Strong Familywise/Maximum Experimentwise	Hochberg, Holm, REGWR, Šidák, Gabriel, Dunnett, DSCF
False Discovery Rate	Benjamini-Hochberg, Student-Newman-Keuls

Selected Type I Error Rate	Possible Parametric Methods
Experimentwise Error Rate	[ANOVA Omnibus Tests,] Protected LSD
Comparisonwise Error Rate	Most Two Sample Tests, Unprotected LSD

After picking which Error Rate you want to control, you choose an appropriate method. There are some pieces of guidance you can follow for choosing the particular method. However, a lot comes down to personal preference.

Checking Appropriateness and Assumptions

Post Hoc analyses involve hypothesis tests. And just like any other hypothesis test, these all have their own set of conditions for whether they are appropriate and for whether we've met their assumptions (note: all methods we're looking at in this course are shortcuts). However, we have a useful advantage.

Since we're doing these tests in a planned, Post Hoc fashion, ***after*** we check the appropriateness of (One-way) ANOVA methods and assess the assumptions of the shortcut, our lives become easy. In essence, if we have satisfied the assumptions for the omnibus shortcuts (ANOVA F test or Kruskal-Wallis H test), then we have satisfied the assumptions for the Post Hoc tests. Similarly, if ANOVA is appropriate, then we know that our Post Hoc methods are appropriate. Thus, taking the time to carefully check appropriateness AND assumptions for the omnibus test pays dividends for us.

Parametric vs. Nonparametric

The last decision we have to make is between using parametric or nonparametric shortcuts. Typically, we want to use parametric shortcuts as they are more powerful (better Type II error control) *provided* their assumptions are satisfied. If you start with a parametric shortcut (i.e., the ANOVA F test), then you need to use a parametric shortcut for the Post Hoc analysis. Similarly, if you end up using the Kruskal-Wallis H test, you need to make use of a nonparametric shortcut.

Post Hoc for the Parametric Shortcut

For Post Hoc analysis, we need to have fitted our ANOVA model. Here is the code and modern ANOVA table for the Cheese Study; I'll leave you to do the same with the Song Knowledge study.

```
cheeseModel <- aov(
  formula = acids ~ strain,
  data = cheeseData,
  na.action = "na.omit"
)

parameters::model_parameters(
  model = cheeseModel,
  omega_squared = "raw",
  eta_squared = "raw",
  epsilon_squared = "raw"
) %>%
  knitr::kable(
  digits = 4,
  col.names = c(
    "Source", "SS", "df", "MS", "F", "p-value",
    "Omega Sq.", "Eta Sq.", "Epslion Sq."),
  caption = "ANOVA Table for Free Amino Acids in Cheese Study",
  booktabs = TRUE,
  align = c("l", rep("c", 8))
  ) %>%
  kableExtra::kable_styling(
    font_size = 10,
    latex_options = c("HOLD_position")
  )
```

Table 2: ANOVA Table for Free Amino Acids in Cheese Study

Source	SS	df	MS	F	p-value	Omega Sq.	Eta Sq.	Epslion Sq.
strain	5.6279	3	1.8760	11.9318	0.0183	0.8039	0.8995	0.8241
Residuals	0.6289	4	0.1572					

We will say for the Cheese Study that we'll reject the null hypothesis and decide to act as if the strain of starter bacteria does impact the level of free amino acids. I will use the Cheese study as my coding example. You should mimic my code not only with the Cheese study (i.e., reproducing my results) but also apply the code to the Song Knowledge study.

Tukey's [& Kramer's] Honest Significant Difference

When you just don't know what to do, controlling the SCI/MEER and using Tukey's or Tukey-Karmer's HSD is a decent default method to use. Keep in mind that if you aren't building confidence intervals, then you are controlling the MEER. The distinction between Tukey's HSD and Tukey-Karmer's HSD is that the former is for exact balanced designs; the later is for imbalanced designs. Both are accessed through the same function, `TukeyHSD` and R will automatically use the correct one.

```
hsdPH <- TukeyHSD(
  x = cheeseModel, # Your aov/lm object
  conf.level = 0.95 # 1 -- Your overall Type I Error level
)

## Kable Code for Tukey HSD
knitr::kable(
  x = hsdPH$strain, # Notice the factor's name
  digits = 3,
  caption = "Post Hoc Tukey HSD Comparisons",
  col.names = c("Difference", "Lower Bound",
                "Upper Bound", "Adj. p-Value"),
  align = 'lcccc',
  booktabs = TRUE,
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 3: Post Hoc Tukey HSD Comparisons

	Difference	Lower Bound	Upper Bound	Adj. p-Value
A-(A & B)	-1.892	-3.506	-0.277	0.030
B-(A & B)	-1.017	-2.631	0.598	0.187
None-(A & B)	-2.137	-3.751	-0.522	0.019
B-A	0.875	-0.739	2.489	0.264
None-A	-0.245	-1.859	1.369	0.921
None-B	-1.120	-2.734	0.494	0.146

You have confidence intervals (Lower Bound, Upper Bound) as well as p-values that are adjusted for the HSD. You can directly compare these to your Unusualness Threshold, $UT \leq \mathcal{E}_I$.

The `TukeyHSD` is part of the base build of R and does not require any special packages.

Pairwise Methods

A second option that is built into the base build of R is that of `pairwise.t.test`. This is an extension of the `t.test` function that you would use for a standard two-sample test for testing location parameters. However, `pairwise.t.test` is built for handling the Multiple Comparison/Simultaneous Inference Problem.

```
pairwisePH <- pairwise.t.test(
  x = cheeseData$acids, # call the response vector
  g = cheeseData$strain, # call your treatment vector
  p.adjust.method = "bonferroni" # Which method you want
)

## Kable Code for Pairwise.t.Test
knitr::kable(
  x = pairwisePH$p.value,
  digits = 3,
  caption = paste( # This creates a nice looking table caption
    "Post Hoc", # feel free to copy
    gsub(
      pattern = "(^|[[:space:]])([[:alpha:]])",
      replacement = "\\1\\U\\2",
      x = pairwisePH$p.adjust.method,
      perl = TRUE
    ),
    "Comparisons"
  ),
  align = rep('c', nrow(pairwisePH$p.value))
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed"),
    font_size = 12,
    latex_options = "HOLD_position"
  ) %>%
  kableExtra::footnote(
    general = "Rows and Columns are Treatment Levels.",
    footnote_as_chunk = TRUE
  )
```

Table 4: Post Hoc Bonferroni Comparisons

	(A & B)	A	B
A	0.053		
B	0.374	0.552	
None	0.034	1.000	0.286

Note: Rows and Columns are Treatment Levels.

The reported p-values are adjusted so that you may compare them against your Usualness Threshold. You can change which method gets used by changing the value of the `p.adjust.method` argument:

Chosen Method	Set `p.adjust.method =`
Bonferroni	`"bonferroni"`
Holm	`"holm"`
Hochberg	`"hochberg"`
Benjamini & Hochberg	`"BH"` OR `"fdr"`

`DescTools` Method

The `DescTools` package also provides a function that will allow you to pairwise comparisons for Post Hoc analysis; `PostHocTest`.

```
## DescTools Pairwise Method
dtPH <- DescTools::PostHocTest(
  x = cheeseModel, # Your aov/lm object
  method = "newmankeuls", # Your chosen method
  conf.level = 0.9 # 1 -- Your Overall Type I Error Rate
)

## Kable Code for DescTools
knitr::kable(
  x = dtPH$strain, # Notice the use of the factor name
  digits = 3,
  caption = paste( # Creates a nice title; copy at will
    "Post Hoc",
    attr(dtPH, "method"),
    "Comparisons"
  ),
  col.names = c("Difference", "Lower Bound",
                "Upper Bound", "Adj. p-Value"),
  align = 'lcccc',
  booktabs = TRUE,
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 6: Post Hoc Newman-Keuls Comparisons

	Difference	Lower Bound	Upper Bound	Adj. p-Value
A-(A & B)	-1.892	-3.006	-0.777	0.019
B-(A & B)	-1.017	-1.862	-0.171	0.062
None-(A & B)	-2.137	-3.422	-0.851	0.019
B-A	0.875	0.030	1.720	0.092
None-A	-0.245	-1.090	0.600	0.570
None-B	-1.120	-2.235	-0.005	0.099

This table is much like the one for Tukey's HSD.

You can change which method gets used by changing the value of the `.method` argument:

Chosen Method	Set `method =`
Tukey HSD	`"hsd"`
Dunn's (Bonferroni)	`"bonf"`
Fisher's Least Significant Difference	`"lsd"`
Scheffé	`"scheffe"`
Newman-Keuls	`"newmankeuls"`

I often prefer using `DescTools` over `pairwise.t.test` as you do get a format for Post Hoc results that is often easier for people to look at and understand.

Special Comparisons (Dunnett's)

In certain studies, you might want to do all possible pairwise comparisons. You might want to compare treatments to a null treatment, a standard care treatment, or the current "gold standard". To make such comparisons, we will need to use the `DescTools` package and Dunnett's Test (`DunnettTest`).

```
# Special Comparisons-Dunnett's Test
dunnett <- DescTools::DunnettTest(
  formula = acids ~ strain,
  data = cheeseData,
  control = "None", # Enter the level that you want to compare to
  conf.level = 0.9 # 1 -- Your Overall Type I Error Rate
)

## Kable Code for Dunnett's Test
knitr::kable(
  x = dunnett$`None`, # Note the use of special treatment level
  digits = 3,
  caption = paste("Post Hoc Comparisons--Dunnett's Test"),
  col.names = c("Difference", "Lower Bound",
                "Upper Bound", "Adj. p-Value"),
  align = 'lcccc',
  booktabs = TRUE
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 8: Post Hoc Comparisons–Dunnett's Test

	Difference	Lower Bound	Upper Bound	Adj. p-Value
(A & B)-None	2.136	0.993	3.280	0.013
A-None	0.245	-0.899	1.389	0.870
B-None	1.120	-0.024	2.264	0.104

Effect Sizes

When you have used the Parametric Shortcuts, you'll want to use the `anova.PostHoc` function from my ANOVATools.R helper file. You'll pass your model object (the result of calling `aov` or `lm`) as the input to this function.

- Note 1: this will generate ALL pairwise comparisons. If you only did a subset, you'll want to remove the extra rows.
- Note 2: some special characters (e.g., &) will get get replaced with periods. I am working on fixing this.

```
anova.PostHoc(cheeseModel) %>%
  knitr::kable(
    digits = 3,
    caption = "Post Hoc Comparison Effect Sizes",
    col.names = c("Pairwise Comparison","Cohen's d", "Hedge's g",
                  "Prob. Superiority"),
    align = 'lccc',
    booktabs = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
```

```
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 9: Post Hoc Comparison Effect Sizes

Pairwise Comparison	Cohen's d	Hedge's g	Prob. Superiority
X.A...B. vs. A	5.443	3.110	1.000
X.A...B. vs. B	2.161	1.235	0.937
X.A...B. vs. None	12.809	7.319	1.000
A vs. B	-1.634	-0.934	0.124
A vs. None	0.803	0.459	0.715
B vs. None	2.545	1.454	0.964

Cohen's d and Hedge's g are both measures of the distance between the *Sample Arithmetic Mean* values for the two groups scaled by pooled standard deviations (Hedge's g uses a weighted pooling). Thus, you can think of these as saying that there are 5.443 standard deviations between the performance for using both A & B strains and just the A strain in terms of free amino acid levels.

The Probability of Superiority measure the percent of the time a randomly selected member of the first group will have the higher numeric value of the response than a randomly selected member of the second group. Thus, we can say that only 12% of the time we randomly select a cheese given just the A strain will that cheese have more free amino acids than a randomly selected cheese given just the B strain.

Post Hoc for the Nonparametic Shortcut

While there are several routes you can take for Nonparametric Post Hoc analysis, there are not as many as for parametric shortcuts. Further, I'm only going to detail two approaches: Dunn's Test and the DSCF Test.

Dunn's Test

Dunn's Test (not to be confused with Dunnett's Test) comes from the `dunn.test` package and provides multiple nonparametric versions of the methods you've already seen.

```
# dunn.test is a bit "noisy" in that it will print
# extraneous output to your console and to your report.
# Use quietly from purrr (part of tidyverse) to stop this

dunn <- purrr::quietly(dunn.test::dunn.test)(
  x = cheeseData$acids, # response vector
  g = cheeseData$strain, # factor vector
  method = "bonferroni", # Your chosen method
  alpha = 0.1, # Your Overall Type I Error Rate
  kw = FALSE, # Turns Off Kruskal Wallis Output
  table = FALSE, # Turns off a default output table
  list = FALSE # Used with step up/down methods
)$result # Don't forget this call to get the result of dunn.test

## Kable Code for Dunn's Test
knitr::kable(
  x = data.frame(
    comparison = dunn$comparisons,
    pvalues = dunn$P.adjusted
  ),
```

```
  digits = 4,
  caption = "Post Hoc Dunn's Test--[insert method here] Adjustment", # Fill this in
  col.names = c("Comparison", "Adj. p-Value"),
  align = 'lc'
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 10: Post Hoc Dunn's Test–[insert method here] Adjustment

Comparison	Adj. p-Value
(A & B) - A	0.1237
(A & B) - B	1.0000
A - B	0.6620
(A & B) - None	0.1237
A - None	1.0000
B - None	0.6620

You can then compare these p-values to your Unusalness Threshold.

You can change which method you use by changing the value of the `method` argument:

Chosen Method	Set `method =`
Bonferroni	`"bonferroni"`
Šidák	`"sidak"`
Holm	`"holm"`
Holm-Šidák	`"hs"`
Hochberg	`"hochberg"`
Benjamini-Hochberg	`"bh"`

For the last four, set `list = TRUE` to have the results be put into the proper ordering and marked for rejection of the null hypothesis.

DSCF Test

The nonparametric counterpart of Tukey's/Tukey-Kramer's HSD is the Dwass-Steel-Critchlow-Fligner All Pairs Test; DSCF Test for short. For this test, you'll want to use the `dscfTest` function from my ANOVATools.R list.

```
# Post Hoc Method Two--DSCF Test
dscf <- dscfTest(
  response = cheeseData$acids,
  factor = cheeseData$strain
)

# Kable Code for DSCF
knitr::kable(
  x = dscf,
  digits = 3,
  col.names = c("Comparison", "Adj. p-value"),
  caption = paste("Post Hoc-Dwass-Steel-Critchlow-Fligner Tests"),
  align = 'lc'
) %>%
```

```
  kableExtra::kable_styling(
    bootstrap_options = c("condensed"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 12: Post Hoc-Dwass-Steel-Critchlow-Fligner Tests

Comparison	Adj. p-value
(A & B) vs. A	0.408
(A & B) vs. B	0.408
(A & B) vs. None	0.408
A vs. B	0.408
A vs. None	1.000
B vs. None	0.408

You would then compare these adjusted p-values to your Unusualness Threshold.

Effect Sizes

When you have used Nonparametric Shortcuts (either through Dunn's Test or the DSCF Test), you'll want to use the `kw.PostHoc` function (also from my helper functions). You'll need to provide two inputs: the response vector and the treatment vector.

```
kw.PostHoc(
    response = cheeseData$acids,
    treatments = cheeseData$strain
  ) %>%
knitr::kable(
  digits = 3,
  caption = "Post Hoc Comparison Effect Sizes",
  col.names = c("Pairwise Comparison","Hodges Lehmann Estimate",
                "Prob. Superiority"),
  align = 'lcc',
  booktabs = TRUE
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )
```

Table 13: Post Hoc Comparison Effect Sizes

Pairwise Comparison	Hodges Lehmann Estimate	Prob. Superiority
(A & B) - A	1.892	1.000
(A & B) - B	1.017	0.736
A - B	-0.875	0.137
(A & B) - None	2.137	1.000
A - None	0.245	0.500
B - None	1.120	0.863

The Probability of Superiority has the same kind of interpretation as in the parametric shortcut: approximately 74% of the time we randomly select a cheese which got both the A & B strains, this cheese will have more free amino acids than a randomly selected cheese which only got the B strain.

The Hodges-Lehmann Estimate, $\widehat{\Delta}$, measures the middle difference of all possible pairings between two groups. If we imagine taking all of the cheeses which got both strains (A & B) and all of the cheeses that got just the A strain, we can imagine all of the pairings of one cheese from each group. For each pairing, we can find the difference in free amino acid levels. For these two groups/treatments, half of all the pairings show that the (A & B) group has at least ~1.8 units more free amino acids.

Visualizing Pairwise Comparisons

When using the parametric shortcuts, you can visualize your pairwise comparisons in a couple of different ways by using the `multcompView` package.

Connecting Letter Report

The Connecting Letters Report is similar to the Oehlert's Underline diagrams (p. 88). Rather than using lines, we use lowercase letters. If different treatments share the same letter, they are statistically indistinguishable from one other. If they have different letters, then there is a statistical difference between those treatments.

```
# Connecting Letters Report
multcompView::multcompLetters4(
  object = cheeseModel, # Your aov/lm object
  comp = hsdPH, # Your stored comparisons
  threshold = 0.1 # Your Overall Type I Error Rate
)
```

```
## $strain
## (A & B)       B       A    None
##     "a"    "ab"     "b"     "b"
```

As you can see from the above raw output, the Connecting Letters report in raw does not look very nice. However, we can improve upon this by joining the Connecting Letters Report with box plots.

Box Plot + Connecting Letters

```
# Box plot with connecting letters
## NOTE: Does not allow you to set a threshold
multcompView::multcompBoxplot(
  formula = acids ~ strain,
  data = cheeseData,
  compFn = "TukeyHSD",
  plotList = list(
    boxplot = list(fig = c(0, 0.85, 0, 1)),
    multcompLetters = list(
      fig = c(0.8, 0.9, 0.1, 0.9),
      fontsize = 12,
      fontface = "bold"
    )
  )
)
```

Play with the values of `fig = c(x1, x2, y1, y2)` to get the best arrangement for your case.

- `x1` is the start proportion of the horizontal (left edge is 0)
- `x2` is then end proportion of the horizontal (right edge is 1)
- `y1` is the start proportion of the vertical (bottom edge is 0)
- `y2` is the end proportion of the vertical (top edge is 1)

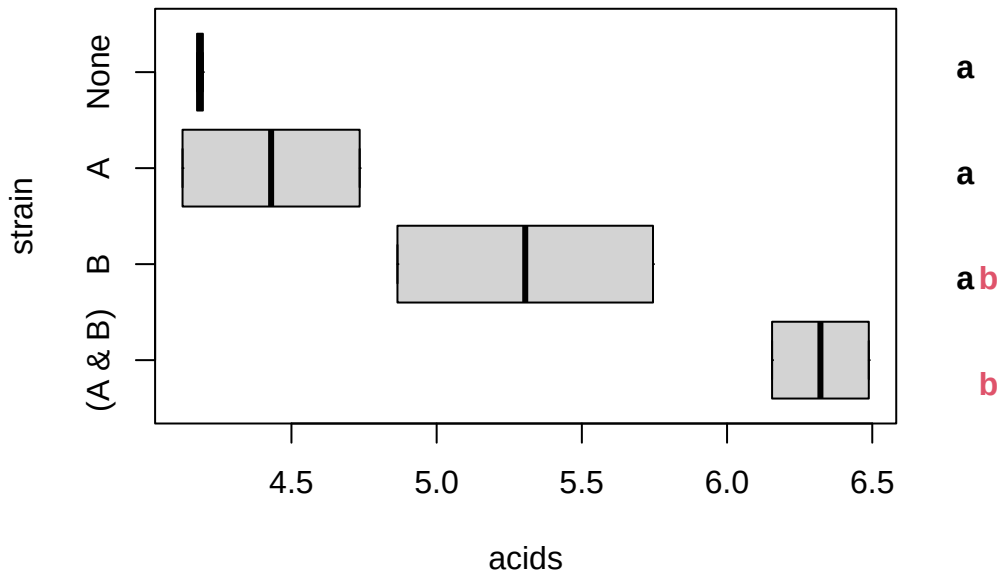

Figure 1: Box Plot and Connecting Letters Report for Free Amino Acids in Cheese Study

Putting Things Together–Your Turn

Use the above examples to explore the Song Knowledge study in terms of Post Hoc Analysis. Generate not only which years in school are significantly different from each other, but what are effect sizes and see if you can re-create the following plot.

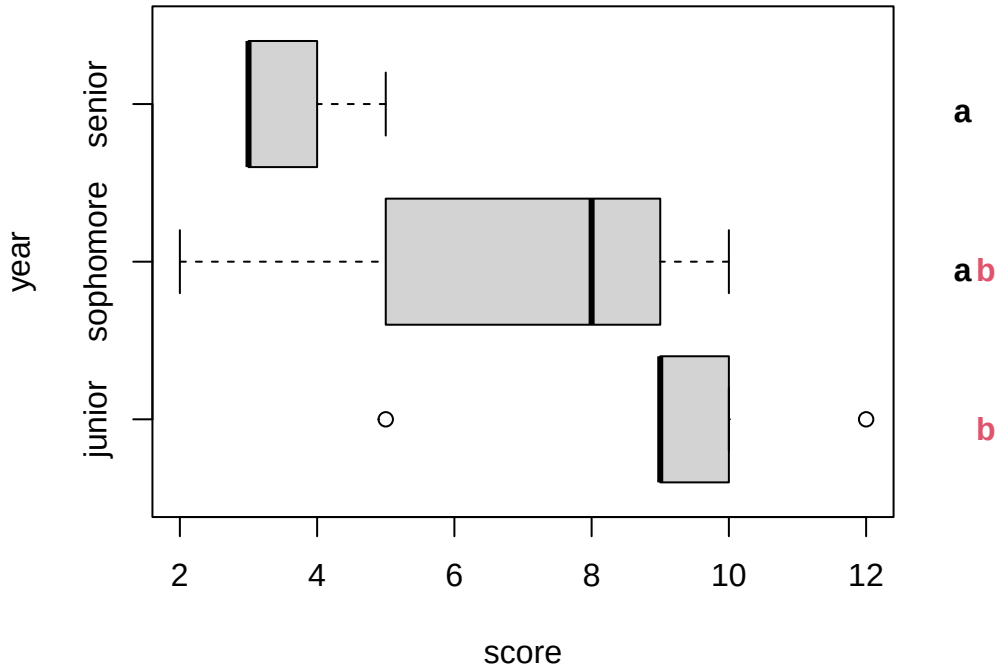

Figure 2: Box Plot and Connecting Letters Report for the Song Knowledge Study

Code Appendix

```
# Setting Document Options
knitr::opts_chunk$set(
  echo = FALSE,
  warning = FALSE,
  message = FALSE,
  fig.align = "center"
)

# Load packages
packages <- c("tidyverse", "knitr", "kableExtra",
              "parameters", "DescTools", "dunn.test", "multcompView")
lapply(packages, library, character.only = TRUE)

# Set options
options(knitr.kable.NA = "")
options(contrasts = c("contr.sum", "contr.poly"))

# Load additional tools
source("https://raw.github.com/neilhatfield/STAT461/master/rScripts/ANOVATools.R")
# Demo code for setting up R
# Load packages
packages <- c("tidyverse", "knitr", "kableExtra",
              "parameters", "DescTools", "dunn.test",
              "PMCMRplus", "multcompView")
lapply(packages, library, character.only = TRUE)

# Set options
options(knitr.kable.NA = "")
options(contrasts = c("contr.sum", "contr.poly"))

# Load additional tools
source("https://raw.github.com/neilhatfield/STAT461/master/rScripts/ANOVATools.R")

# Demo code for loading data
# Song Knowledge study
songData <- read.csv(
  file = "https://raw.github.com/neilhatfield/STAT461/master/dataFiles/songKnowledge2022.csv",
  header = TRUE,
  sep = ","
)
# Set year to an ordered factor
songData$year <- factor(
  x = songData$year,
  levels = c("sophomore", "junior", "senior")
)

# Cheese study
# Create the Cheese data frame
cheeseData <- data.frame(
  strain = as.factor(
    c("None", "None", "A", "A",
      "B", "B", "(A & B)", "(A & B)")
  ),
  acids = c(
    4.195, 4.175, 4.125, 4.735,
    4.865, 5.745, 6.155, 6.488
  )
```

```
)

# Notice that I used as.factor in the data.frame call;
# this is another approach you can use

cheeseModel <- aov(
  formula = acids ~ strain,
  data = cheeseData,
  na.action = "na.omit"
)

parameters::model_parameters(
  model = cheeseModel,
  omega_squared = "raw",
  eta_squared = "raw",
  epsilon_squared = "raw"
) %>%
  knitr::kable(
  digits = 4,
  col.names = c(
    "Source", "SS", "df", "MS", "F", "p-value",
    "Omega Sq.", "Eta Sq.", "Epslion Sq."),
  caption = "ANOVA Table for Free Amino Acids in Cheese Study",
  booktabs = TRUE,
  align = c("l", rep("c", 8))
  ) %>%
  kableExtra::kable_styling(
    font_size = 10,
    latex_options = c("HOLD_position")
  )

hsdPH <- TukeyHSD(
  x = cheeseModel, # Your aov/lm object
  conf.level = 0.95 # 1 -- Your overall Type I Error level
)

## Kable Code for Tukey HSD
knitr::kable(
  x = hsdPH$strain, # Notice the factor's name
  digits = 3,
  caption = "Post Hoc Tukey HSD Comparisons",
  col.names = c("Difference", "Lower Bound",
                "Upper Bound", "Adj. p-Value"),
  align = 'lcccc',
  booktabs = TRUE,
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

pairwisePH <- pairwise.t.test(
  x = cheeseData$acids, # call the response vector
  g = cheeseData$strain, # call your treatment vector
  p.adjust.method = "bonferroni" # Which method you want
)

## Kable Code for Pairwise.t.Test
```

```
knitr::kable(
  x = pairwisePH$p.value,
  digits = 3,
  caption = paste( # This creates a nice looking table caption
    "Post Hoc", # feel free to copy
    gsub(
      pattern = "(^|[[:space:]])([[:alpha:]])",
      replacement = "\\1\\U\\2",
      x = pairwisePH$p.adjust.method,
      perl = TRUE
    ),
    "Comparisons"
  ),
  align = rep('c', nrow(pairwisePH$p.value))
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed"),
    font_size = 12,
    latex_options = "HOLD_position"
  ) %>%
  kableExtra::footnote(
    general = "Rows and Columns are Treatment Levels.",
    footnote_as_chunk = TRUE
  )

## DescTools Pairwise Method
dtPH <- DescTools::PostHocTest(
  x = cheeseModel, # Your aov/lm object
  method = "newmankeuls", # Your chosen method
  conf.level = 0.9 # 1 -- Your Overall Type I Error Rate
)

## Kable Code for DescTools
knitr::kable(
  x = dtPH$strain, # Notice the use of the factor name
  digits = 3,
  caption = paste( # Creates a nice title; copy at will
    "Post Hoc",
    attr(dtPH, "method"),
    "Comparisons"
  ),
  col.names = c("Difference", "Lower Bound",
                "Upper Bound", "Adj. p-Value"),
  align = 'lcccc',
  booktabs = TRUE,
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

# Special Comparisons-Dunnett's Test
dunnett <- DescTools::DunnettTest(
  formula = acids ~ strain,
  data = cheeseData,
  control = "None", # Enter the level that you want to compare to
  conf.level = 0.9 # 1 -- Your Overall Type I Error Rate
)
```

```
## Kable Code for Dunnett's Test
knitr::kable(
  x = dunnett$`None`, # Note the use of special treatment level
  digits = 3,
  caption = paste("Post Hoc Comparisons--Dunnett's Test"),
  col.names = c("Difference", "Lower Bound",
                "Upper Bound", "Adj. p-Value"),
  align = 'lcccc',
  booktabs = TRUE
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

anova.PostHoc(cheeseModel) %>%
  knitr::kable(
    digits = 3,
    caption = "Post Hoc Comparison Effect Sizes",
    col.names = c("Pairwise Comparison","Cohen's d", "Hedge's g",
                  "Prob. Superiority"),
    align = 'lccc',
    booktabs = TRUE
  ) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

# dunn.test is a bit "noisy" in that it will print
# extraneous output to your console and to your report.
# Use quietly from purrr (part of tidyverse) to stop this

dunn <- purrr::quietly(dunn.test::dunn.test)(
  x = cheeseData$acids, # response vector
  g = cheeseData$strain, # factor vector
  method = "bonferroni", # Your chosen method
  alpha = 0.1, # Your Overall Type I Error Rate
  kw = FALSE, # Turns Off Kruskal Wallis Output
  table = FALSE, # Turns off a default output table
  list = FALSE # Used with step up/down methods
)$result # Don't forget this call to get the result of dunn.test

## Kable Code for Dunn's Test
knitr::kable(
  x = data.frame(
    comparison = dunn$comparisons,
    pvalues = dunn$P.adjusted
  ),
  digits = 4,
  caption = "Post Hoc Dunn's Test--[insert method here] Adjustment", # Fill this in
  col.names = c("Comparison", "Adj. p-Value"),
  align = 'lc'
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
```

```
    latex_options = "HOLD_position"
  )
# Post Hoc Method Two--DSCF Test
dscf <- dscfTest(
  response = cheeseData$acids,
  factor = cheeseData$strain
)

# Kable Code for DSCF
knitr::kable(
  x = dscf,
  digits = 3,
  col.names = c("Comparison", "Adj. p-value"),
  caption = paste("Post Hoc-Dwass-Steel-Critchlow-Fligner Tests"),
  align = 'lc'
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

kw.PostHoc(
    response = cheeseData$acids,
    treatments = cheeseData$strain
  ) %>%
knitr::kable(
  digits = 3,
  caption = "Post Hoc Comparison Effect Sizes",
  col.names = c("Pairwise Comparison","Hodges Lehmann Estimate",
                "Prob. Superiority"),
  align = 'lcc',
  booktabs = TRUE
) %>%
  kableExtra::kable_styling(
    bootstrap_options = c("condensed", "boardered"),
    font_size = 12,
    latex_options = "HOLD_position"
  )

# Connecting Letters Report
multcompView::multcompLetters4(
  object = cheeseModel, # Your aov/lm object
  comp = hsdPH, # Your stored comparisons
  threshold = 0.1 # Your Overall Type I Error Rate
)

# Box plot with connecting letters
## NOTE: Does not allow you to set a threshold
multcompView::multcompBoxplot(
  formula = acids ~ strain,
  data = cheeseData,
  compFn = "TukeyHSD",
  plotList = list(
    boxplot = list(fig = c(0, 0.85, 0, 1)),
    multcompLetters = list(
      fig = c(0.8, 0.9, 0.1, 0.9),
      fontsize = 12,
      fontface = "bold"
```

```
    )
  )
)

# Your Turn Code-----------------------------------
multcompView::multcompBoxplot(
  formula = score ~ year,
  data = songData,
  compFn = "TukeyHSD",
  plotList = list(
    boxplot = list(fig = c(0, 0.85, 0, 1)),
    multcompLetters = list(
      fig = c(0.8, 0.9, 0.1, 0.9),
      fontsize = 12,
      fontface = "bold"
    )
  )
)
```