Analysis of Variance

http://datascience.tntlab.org

Module 8

Today's Agenda

- Why were these Data Camp courses so different from the others?

- Stepping back through Data Camp material on **aov**(), sort of
  - How to run t-tests
  - How to run ANOVA's, with some new quick-reference guides
  - Additional visualization tools for ANOVA
  - Bridging what you already learned and *tidyverse*

- *Warning*: Introducing a lot of packages today
  - A lot more material *not in Data Camp* than usual
  - Summaries at the end

A Bit Different Data Camp

- Did not take a *tidyverse* approach; lessons were back in base-R
- We don't like to use base-R if we can avoid it
- You see how weird and ugly figure creation is
- We try to avoid base-R (mostly) but sometimes it is not possible

- Why did this happen?
  - ANOVA is specific type of regression anyway (i.e., the general linear model)
  - ANOVA is very uncommon in the broader data science space because it is very limited
  - Most data science folks don't learn ANOVA because they're going to learn regression anyway, and the general linear model is much more flexible

  - Psychology (and most other social sciences) are trapped in the 20th-century re: statistical methods (1888: correlation, 1908: t-tests, 1918: variance, 1921: ANOVA)
  - Most of these methods became popular due to computational simplicity

The *psych* Package

- A set of functions for doing "traditional" psychological tests and analyses
- Includes both CTT and IRT functions
- Includes tools for many traditional psych problems, including factor analysis, ANOVA, item scoring, commonly needed graphs, data simulation, etc.

- The philosophy of *psych* is very clearly:
  "here's what people commonly want to do in SPSS or SAS"

- We'll dip into *psych* a few times today; it's important you know where each function is located; we'll need it more later
  - If you already have a function loaded but don't remember from where, use **find()**

- One handy function in psych is **describe**()
- We will also be using a dataset included in *psych:* **sat.act**

We Skipped T-Tests

- Because they are code-wise almost the same as ANOVA;
  still uses formula notation

- **t.test(y~x)** # independent-samples t where y is the DV and x is the IV
- **t.test(y1, y2)** # independent-samples t test where y1 is one group's DV
  # scores and y2 is the other's
- **t.test(y, mu=0)** # one-sample t-test with specified mu

- Modifications
  - **t.test(y1, y2, paired = T)** # paired-samples t
  - **t.test(y~x, alternative = "greater")** # one-sided t
  - **t.test(y~x, var.equal = T)** # what SPSS does; R defaults to Welch
  - **t.test(y~x, conf.level = 0.9)** # sets alpha explicitly (as 1-alpha)

The ANOVA Framework

- **aov()** == ANOVA, part of base-R's "*stats* package"
  - If you need something you haven't used before, try looking at **library(help="stats")**

- Also uses formula notation
  - y ~ x == y on x
  - y ~ x1 + x2 == y on x1 and x2
  - y ~ x1 * x2 == y on x1, x2, and their interaction
  - y ~ x1 + x2 + x1:x2 == same thing

- Homogeneity of variance checks using the *car* package (not base-R)
  - **leveneTest(y, x, center = mean)** or **leveneTest(y~x, center = mean)**
- Sphericity using **mauchly.test**()
- Normality checks using **ggplot/ggpairs** and/or *psych*'s **describeBy**() plus **shapiro.test**()
- Independence assumption should be safe given your research design

A Quick and Dirty Assumption-Checking Visualization

- *yarrr* library
  - Quickly creates a type of RDI plot: Raw data, Description, Inference
  - Useful for checking assumptions when modeling a DV on one to three IVs

  - Raw data
    - individual jittered points representing raw data
  - Description
    - a thick line representing the center of the data (by default, the median)
    - a smoothed density plot (by default, a violin plot) representing data density
  - Inference
    - a rectangle representing an inferential interval (by default, a Bayesian Highest Density Interval, i.e., a credibility interval)

  - Only can be used for up to 3 IVs
  - Example
    - **pirateplot(formula = SATQ ~ gender, data = sat.act)**
    - **pirateplot(formula = SATQ ~ gender, data = sat.act, inf.method="ci")**

Fundamental ANOVA Designs with **aov**()

- Between-subjects
  - One-way: **model <- aov(y ~ b1)**
  - Two-way: **aov(y ~ b1 * b2)** –or– **aov(y ~ b1 + b2 + b1:b2)**
  - Three-way: **aov(y ~ b1 * b2 * b3)**
  - With blocking / order effects / covariates: **aov(y ~ z + b1 * b2)**
- Within-subjects (be sure to restructure so that each measurement is a row)
  - One factor: **aov(y ~ w1 + Error(SubjectID / w1))**
  - Two factor: **aov(y ~ w1 * w2 + Error(SubjectID / (w1 + w2))**
- Mixed model
  - One-way + one factor: **aov(y ~ w1 * b1 + Error(SubjectID / w1))**
  - Two-way + one factor: **aov(y ~ w1 * b1 * b2 + Error(SubjectID / w1) + b1 * b2)**
  - Two-way + two factor: **aov(y ~ w1*w2*b1*b2 + Error(SubjectID / (w1+w2)) + b1*b2)**

- You can also get a variety of diagnostic plots with **plot(model)**
- If you don't get full ANOVA tables, you probably mis-specified something

Bridging *tidyverse* and **aov()** for Input

- Between-subjects is easy; just clean as normal
- For within-subjects, you usually have data in the following format:
  - One row per case
  - One column per observation within person
- What you need is:
  - One row per observation
  - One identifier column and one data column
- So, to use within-subject **aov()**, you need to restructure using **gather()**
  - Remember this is from *tidyr* (but don't call *tidyr*)

- From a dataset containing 4 observations and a participant number
  - **original <- tibble(casenum=1:100, time1=rnorm(100, 3, .75), time2=rnorm(100, 3, .75), time3=rnorm(100, 3, .75), time4=rnorm(100, 3, .75))**
  - **new <- gather(original, time, score, time1:time4)**

Post-hoc ANOVA Analyses

- Post hoc tests with **pairwise.t.test(y, x, p.adjust="type")**
  - Types are holm, hochberg, hommel, Bonferroni, BH or fdr, BY
  - Remember to look at documentation to double-check what these do, specifically

  - **pairwise.t.test(sat.act$SATQ, factor(sat.act$education), p.adjust.method="bonferroni")** # add paired=T for w/i
    # –or–
  - **with(sat.act, pairwise.t.test(SATQ, factor(education), p.adjust.method="bonferroni"))**

- Use **TukeyHSD(model)** for basic Tukey post-hoc tests (but this does not work with certain more complex models)

- Effect size calculating using **etaSquared()** from the *lsr* package
  - **etaSquared(sat_aov)**
  - **etaSquared(sat_aov, anova=T)** # for eta-sq plus full ANOVA summary table

More Flexible Post-hoc Testing

- If you need post-hoc testing that **TukeyHSD**() won't do, you may need **glht**() from the *multcomp* package, which is much more flexible
  - **my_df <- sat.act %>% transmute(education = as.factor(education), gender = as.factor(gender), SATQ)**
  - **model <- aov(SATQ ~ education, data=my_df)**
  - **posthocs <- glht(model, linfct=mcp(education="Tukey"))**
  - **summary(posthocs)**

- For post-hocs with multi-way models, you need to define the level condition intersection points of interest yourself, which adds two steps
  - **my_df <- sat.act %>% transmute(education = as.factor(education), gender = as.factor(gender), SATQ, ACT)**
  - **model <- aov(SATQ ~ ACT + education * gender, data=my_df)**
  - **my_df %<>% mutate(condition = interaction(education, gender, sep="x"))**
  - **posthoc_model <- aov(SATQ ~ ACT + condition, data=my_df)**
  - **posthocs <- glht(posthoc_model, linfct=mcp(condition="Tukey"))**
  - **summary(posthocs)**

Bridging *tidyverse* and **aov()** for Output

- Use the *broom* package's **tidy** function
  - Works on many outputs, e.g., model specifications, t-tests
  - Don't use it on the **summary**() output; use it on the model variable
  - Use this as the first step in a pipe for visualization or further analyses

- Compare
  - **model <- aov(SATQ~factor(education), data=sat.act)**
  - **summary(model)**
  - **tidy(model)**

For Publication-ready Output

- *apaTables* library
  - Creates APA style tables with expected decimal places from various summary tables for MS Word
  - Be sure all variables have the correct type ahead of time
  - Compare
    - **summary(model)**
    - **apa.aov.table(model, conf.level=.95)**
    - **apa.aov.table(model, "../output/output.doc", conf.level=.95)**
  - For cell descriptives, use **apa.1way.table**() or **apa.2way.table**()
    - **apa.1way.table(iv=education, dv=SATQ, data=sat.act, filename="output.doc")**
  - For ANOVA paired comparisons, use **apa.d.table**() to get effect sizes matrix
    - **apa.d.table(iv=education, dv=ACT, data=sat.act, filename="output.doc")**

Some Added Complexity

- If you have **multiple predictors** and especially **correlated predictors**, you probably need to worry about the way sum of squares is calculated
- aov() uses **Type I Sum of Squares** ("sequential"), which uses sequential entry of terms into the model
  - **aov(y ~ x1 * x2)** != **aov(y ~ x2 * x1)**
- You usually shouldn't default to Type I unless you're specifically interested in sequential, incremental prediction (or only the last term in the model)

- A **Type II Sum of Squares** ("hierarchical") controls for main effects but adds interaction sequentially
  - Main effects do not control for interactive effects
- **A Type III Sum of Squares** ("marginal" or "orthogonal") controls for all other effects in the model
  - Main effects are uninterpretable in the presence of an interaction
  - This is the default you're used to if you learned in SPSS

- If you design is *balanced*, i.e., all predictors are uncorrelated, these will be equal

Type II or III SS for ANOVA

- To run ANOVA with Type II (or III) SS, you need to run the general linear model instead and then wrap it in the **Anova**() function from *car* (*note*. **aov**() is a wrapper around **lm**() too)

- Instead of this: **anova_model <- aov(y ~ x1 * x2)**
- Do this: **options(contrasts = c("contr.sum", "contr.poly"))**
  (first line only once) **linear_model <- lm(y ~ x1 * x2)**
  anova_model <- Anova(linear_model, type="III") # notice capital

- Compare
  - **my_act <- sat.act %>% transmute(ACT, age, gender=factor(gender), education=factor(education))**
  - **summary(aov(ACT ~ gender * education, data=my_act))**
  - **summary(aov(ACT ~ gender * education, data=my_act))**
  - **Anova(lm(ACT ~ gender * education, data=my_act), type="III")**
  - **Anova(lm(ACT ~ gender * education, data=my_act), type="III")**

Other **aov**() > **lm**() Changes

- Remember from earlier: **aov(y ~ z + b1 * b2)**

- Most helper functions require the **lm**, but it can be a bit weird

  - **etaSquared(linear_model, type=3)** # don't forget the type parameter

  - **apa.aov.table(linear_model, type=3)** # same problem

  - **library(mosaic)**
    TukeyHSD(linear_model)
    # Tukey will throw an error without
    # this library loaded; it wraps itself

Marginal Means Plots

- Use the *lsmeans* library, which creates least-squares means (i.e., predicted marginal means) for any linear model; can be used with any **lm**

- All functions take a model plus the piece of the model you want to look at marginal means for

- **lsmeans(model)** creates marginal ("least-square") means based upon the part of the model you want to visualize or report
  - **linear_model <- lm(ACT ~ gender * education + age, data=my_act)**
  - **anova_model <- Anova(linear_model, type="III")**
  - **lsmeans(linear_model, "education")** # on original linear model, not Anova
    # use these means as data for ggplot()

Planned Contrasts

- Also from the *lsmeans* package
  - **contrast(model, list(contrast_name = c(-1,1))**
  - Does not matter if you use **aov**() or **lm**(), because this is a planned test (SS "type" is not relevant when you specify precisely what to compare to what)
  - You might need **glht** from *multcomp* depending upon your model

- Example
  - **model <- aov(ACT ~ education, data=my_act)**
  - **model.lsm <- lsmeans(model, "education")**
  - **contrast(model.lsm, list(**
    hi_v_lo = c(-1,-1,0,0,1,1), # 2 highest v 2 lowest
    max_v_min = c(-1,0,0,0,0,1) # highest v lowest
)
)

Example Workflow

1. my_act <- sat.act %>% transmute(ACT, age, gender=factor(gender), education=factor(education))
2. options(contrasts = c("contr.sum", "contr.poly"))
3. linear_model <- lm(ACT ~ age + gender * education, data=my_act)
4. anova_model <- Anova(linear_model, type="III")
5. anova_model
6. my_act %<>% mutate(condition = interaction(education, gender, sep="x"))
7. posthoc_model <- aov(ACT ~ age + condition, data=my_act)
8. posthocs <- glht(posthoc_model, linfct=mcp(condition="Tukey"))
9. summary(posthocs)
10. mm_df <- tidy(lsmeans(linear_model, "gender", by="education"))
11. ggplot(mm_df, aes(x=gender, y=estimate, ymax=conf.high, ymin=conf.low, color=education)) + geom_errorbar(position="dodge")
8. ggplot(mm_df, aes(x=gender, y=estimate, group=education, color=education)) + geom_line()
9. apa.aov.table(linear_model, conf.level=.95, filename="my_aov.doc")

Summary of Libraries, Functions, and Datasets

- Base-R: **summary()**, **interaction()**
- Base-R/*stats*: **aov()**, **t.test()**, **mauchly.test()**, **TukeyHSD()**, **shapirpairwise.t.test()**, **lm()**, **options()**
- Base-R/*utils*: **citation()**, **find()**
- *tidyverse/tidyr*: **gather()**
- *tidyverse/dplyr*: **transmute()**
- *tidyverse/ggplot2*: **ggplot()**
- *apaTables*: **apa.aov.table()**, **apa.1way.table()**, **apa.2way.table()**, **apa.d.table()**
- *broom*: **tidy()**
- *car*: **leveneTest()**, **Anova()**
- *lsmeans*: **lsmeans()**, **contrast()**
- *lsr*: **etaSquared()**
- *mosaic*: **TukeyHSD()** # wrapper around Base/*stats* TukeyHSD for use on **lm**() output
- *multcomp*: **glht()**
- *psych*: **describe()**, **describeBy()**, **bfi**, **sat.act**
- *yarrr*: **pirateplot()**