R Workshop 7: Tables

Teal Potter

10/22/2021

Table of Contents

Setup 1

Save simple stats output  3

Save table as a .csv or .txt file 3

Extract statistics from saved model 4

Build a customized statistics table  5

R Markdown options  6

If you are testing more than 1 hypothesis using the same statistical test then including the useful statistics in a table is often cleaner than including statistics in parentheses in your results section text in a manuscript. Here are some ways to avoid errors in manually transcribing values from your model outputs in R into tables in a Word document.

Setup

```
library(car) # contains Soils dataset
library(knitr)
library(flextable)

head(Soils)
```

```
##   Group Contour Depth Gp Block   pH     N Dens   P    Ca   Mg    K   Na Conduc
## 1     1     Top  0-10 T0     1 5.40 0.188 0.92 215 16.35 7.65 0.72 1.14   1.09
## 2     1     Top  0-10 T0     2 5.65 0.165 1.04 208 12.25 5.15 0.71 0.94   1.35
## 3     1     Top  0-10 T0     3 5.14 0.260 0.95 300 13.02 5.68 0.68 0.60   1.41
## 4     1     Top  0-10 T0     4 5.14 0.169 1.10 248 11.92 7.88 1.09 1.01   1.64
## 5     2     Top 10-30 T1     1 5.14 0.164 1.12 174 14.17 8.12 0.70 2.17   1.85
## 6     2     Top 10-30 T1     2 5.10 0.094 1.22 129  8.55 6.92 0.81 2.67   3.18
```

First, let's run some ANOVAs and build a statistical table with appropriate statistics we should report.

Example anova model: Does N vary by depth?

```
anova1 <- anova(aov(Soils$N ~ Soils$Depth))

anova1

## Analysis of Variance Table
##
## Response: Soils$N
##             Df   Sum Sq  Mean Sq F value    Pr(>F)
## Soils$Depth  3 0.163714 0.054571  49.745 3.465e-14 ***
## Residuals   44 0.048269 0.001097
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

regression1 <- summary(lm(Soils$pH ~ Soils$N))
```

First, it's good to know a few functions to get a view of what is saved in the models you run and save to an object.

```
ls(anova1)

## [1] "Df"      "F value" "Mean Sq" "Pr(>F)"  "Sum Sq"

ls(regression1)

##  [1] "adj.r.squared" "aliased"       "call"          "coefficients"
##  [5] "cov.unscaled"  "df"            "fstatistic"    "r.squared"
##  [9] "residuals"     "sigma"         "terms"

attributes(anova1)

## $names
## [1] "Df"      "Sum Sq"  "Mean Sq" "F value" "Pr(>F)"
##
## $row.names
## [1] "Soils$Depth" "Residuals"
##
## $class
## [1] "anova"      "data.frame"
##
## $heading
## [1] "Analysis of Variance Table\n" "Response: Soils$N"

attributes(regression1)

## $names
##  [1] "call"          "terms"         "residuals"     "coefficients"
##  [5] "aliased"       "sigma"         "df"            "r.squared"
##  [9] "adj.r.squared" "fstatistic"    "cov.unscaled"
##
```

```
## $class
## [1] "summary.lm"
```

Save simple stats output  EASIEST

Tidy outputs in R script using the tidy function in the broom package. This function also turns the model output into a data frame. First I'll print the standard output so you can compare to the tidy version.

```
#install.packages("broom")
library(broom)

anova1

## Analysis of Variance Table
##
## Response: Soils$N
##             Df   Sum Sq  Mean Sq F value    Pr(>F)
## Soils$Depth  3 0.163714 0.054571  49.745 3.465e-14 ***
## Residuals   44 0.048269 0.001097
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

broom::tidy(anova1)

## # A tibble: 2 × 6
##   term           df  sumsq  meansq statistic   p.value
##   <chr>       <int>  <dbl>   <dbl>     <dbl>     <dbl>
## 1 Soils$Depth     3 0.164  0.0546       49.7  3.46e-14
## 2 Residuals      44 0.0483 0.00110      NA   NA

broom::tidy(regression1)

## # A tibble: 2 × 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     4.02     0.138     29.0  3.10e-31
## 2 Soils$N         6.37     1.14       5.60 1.15e- 6
```

This is a simple option for saving your stats table so you can access it outside of your script it to save this tidy data frame as a .csv or .txt file.

Save table as a .csv or .txt file

The syntax for write.csv(x = data object name, file = "path to where you want to save on your computer/give your file a name.csv")

Then you can copy/paste your table fro Excel into a table in a Word doc and use Word's formatting tools to further include italics, symbmols, horizontal lines, etc.

```
table1 <- tidy(anova1)

write.csv(x = table1, file = "/Users/tealpotter/OneDrive WSU/R workshop Series/R wk7 Tables/study1_ANOVA_table_depth_vs_N.csv", row.names = FALSE)
```

Alternatively, you can save a text file as an intermediate step to importing the data into a table in word. Here are some instructions: https://sejdemyr.github.io/r-tutorials/basics/tables-in-r/

```
write.table(x = table1, file = "/Users/tealpotter/OneDrive WSU/R workshop Series/R wk7 Tables/study1_ANOVA_table_depth_vs_N.txt", sep = ",", quote = FALSE, row.names = FALSE)
```

Often times the stats output include stats that are not needed for a stats table in a publication and sometimes the default calculations are not the most appropriate to report for your data. Thus, it may be more practical to build your own statistics table from scratch. To do this you need to know how to access and save specific values from the model output. Here are are a couple ways to do this:

Extract statistics from saved model

```
anova1[1,1] # extract df value: row 1, column 1 of output table

## [1] 3

anova1$`Df`[1] # extract df value by column name, row 1

## [1] 3

coef(regression1)

##             Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 4.020126  0.1384126 29.044510 3.100081e-31
## Soils$N     6.369092  1.1374538  5.599429 1.148823e-06

coefficients(regression1) # same coef()

##             Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 4.020126  0.1384126 29.044510 3.100081e-31
## Soils$N     6.369092  1.1374538  5.599429 1.148823e-06

regression1$coefficients

##             Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 4.020126  0.1384126 29.044510 3.100081e-31
## Soils$N     6.369092  1.1374538  5.599429 1.148823e-06

regression1$coefficients[1,1]

## [1] 4.020126
```

Build a customized statistics table INTERMEDIATE

Here, I place the stats I want into a data frame using column names to extract statistics as an example. First, I'll show the ANOVA output again. Highlighted colors in model output match table elements. Note that the stats included here are probably not sufficient for a publishable ANOVA table.

```
anova1
```

```
Analysis of Variance Table

Response: Soils$N
             Df   Sum Sq   Mean Sq F value   Pr(>F)
Soils$Depth   3 0.163714 0.054571  49.745 3.465e-14***
Residuals    44 0.048269 0.001097
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
row1 <- data.frame(
  Model = "Depth",
  df    = anova1$`Df`[1],
  SS    = anova1$`Sum Sq`[1],
  Fval  = anova1$'F value'[1],
  P     = anova1$'Pr(>F)'[1])

row1
```

```
##   Model df        SS     Fval            P
## 1 Depth  3 0.1637141 49.74522 3.464563e-14
```

Here is a slightly cleaner version of the same table where numbers have been rounded and p-values have been changed to <0.01 if they were rounded to 0.

```
row1 <- data.frame(
  Model = "Depth",
  df    = anova1$`Df`[1],
  SS    = round(anova1$`Sum Sq`[1], 3),
  Fval  = round(anova1$'F value'[1], 2),
  P     = ifelse(test = round(anova1$'Pr(>F)'[1], 3) == 0, yes = "< 0.01", no =
round(anova1$'Pr(>F)'[1], 3)))

row1
```

```
##   Model df    SS  Fval      P
## 1 Depth  3 0.164 49.75 < 0.01
```

Now I'm going to run a new ANOVA to see if N varies by sampling blocks to make a second row for our our stats tables.

Note: if you want to run a bunch of similar tests it may be worth investing in learning how to write functions, for loops, or new tools in the broom package.

```
anova2 <- anova(aov(Soils$N ~ Soils$Block))

row2 <- data.frame(   # changed to row2
  Model = "Block",    # changed to Block
  df    = anova2$`Df`[1],  # changed to anova2 here and below
  SS    = round(anova2$`Sum Sq`[1], 3),
  Fval  = round(anova2$'F value'[1], 2),
  P     = ifelse(test = round(anova2$'Pr(>F)'[1], 3) == 0, yes = "< 0.01", no =
round(anova2$'Pr(>F)'[1], 3)))
```

Stack your rows that represent two different statistical tests together in the same table

```
table2 <- rbind(row1,row2)

table2

##   Model df    SS  Fval      P
## 1 Depth  3 0.164 49.75 < 0.01
## 2 Block  3 0.004  0.27  0.847

flextable::flextable(rbind(row1, row2))
```

Model	df	SS	Fval	P
Depth	3	0.164	49.75	< 0.01
Block	3	0.004	0.27	0.847

Note that you can add the function autofit() for wide tables to ask R to try and fit everything nicely. E.g. autofit(flextable(table1))

R Markdown options 

Tidy outputs in R Markdown

```
knitr::kable(table2)
```

Model	df	SS	Fval	P
Depth	3	0.164	49.75	< 0.01
Block	3	0.004	0.27	0.847

Looks good. If you are using R Markdown and saving to a Word doc, flextable() offers more functionality than kable() to customize your table before exporting (aka rendering) into a word document.

If you are creating reports in R markdown and saving to a pdf, you can use kable() and extra options for nice looking tables with striped rows for example if you are ok with rendering to a pdf instead of a Word doc. Here is a good resource on customizing tables using knitr::kable and kableExtra packages.
https://www.youtube.com/watch?v=JqUViTDoSEo

```
t.formatted <- kable(table2, booktabs = T) # %>%

kable_styling(t.formatted, latex_options = "striped")
```

Model	df	SS	Fval	P
Depth	3	0.164	49.75	< 0.01
Block	3	0.004	0.27	0.847