EDUC 640

Two–Way ANOVA

Chris Ives

Contents

- Foundations
  - Crosstabs
  - Clustered Boxplots
- ANOVA
  - Two–way ANOVA
  - Marginal Means
  - Simple Effects
  - Interaction Plots

PSA – Scientific Notation

Sometimes I find scientific notation to be a bit annoying to work with in R. If you want to turn it off for your session run the following code:

```
options(scipen = 999)
```

For the sake of table formatting, I'll keep mine on for the slides.

Crosstabs

The following code shows how to make crosstabs using `dplyr` functions. `group_by` and `summarise` are good to know, so I'd suggest running these lines one by one so you can get a sense of what they're doing. `summarise` creates a row for each group we've specified, then we specify the column contents.

```
data %>%
  group_by(inst, meth) %>%
  summarise(n = n()) %>% #new column "n" = row count of each fac
  spread(meth, n)
```

```
## `summarise()` has grouped output by 'inst'. You can override using the `
```

```
## # A tibble: 3 x 3
## # Groups:   inst [3]
##   inst              computer standard
##   <fct>                <int>    <int>
## 1 physical science         6        6
## 2 social science           6        6
```

Check Descriptives

```
rmarkdown::paged_table(
  describe(data)
  )
```

	vars	n	mean	sd
	<int>	<dbl>	<dbl>	<dbl>
idnum	1	36	18.5	10.5356538
vocab	2	36	33.5	12.8652355
inst*	3	36	2.0	0.8280787
meth*	4	36	1.5	0.5070926

4 rows | 1–5 of 14 columns

Grouped Descriptives

Setting `mat = TRUE` gives you the output for each level in one data frame.

```
rmarkdown::paged_table(
  describeBy(x = data$vocab, group = data$inst,
             mat = TRUE, data = data)
  )
```

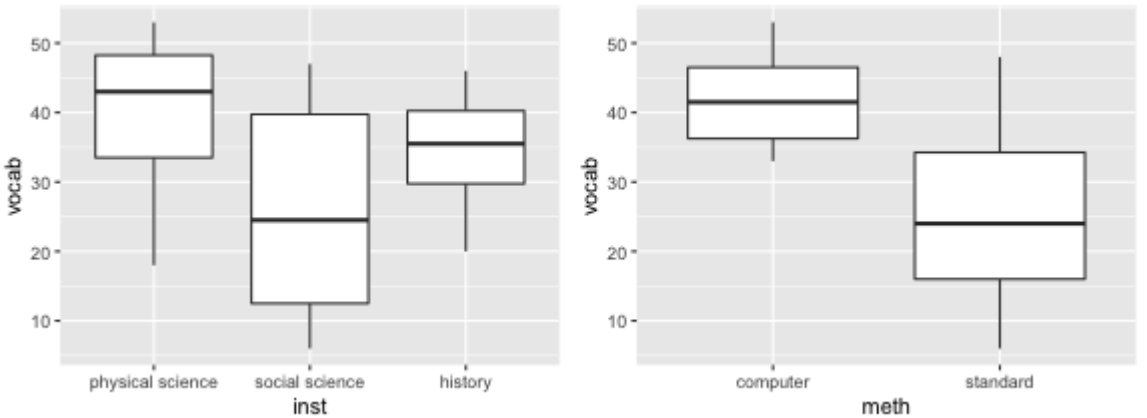
	item	group1	vars	n
	<chr>	<chr>	<dbl>	<dbl>
X11	1	physical science	1	12
X12	2	social science	1	12
X13	3	history	1	12

3 rows | 1–5 of 16 columns

Separate Boxplots

Since we already covered normal boxplots, I'll just demonstrate a way to plot multiple boxplots if the need every arises. You'll need the `gridExtra` package.

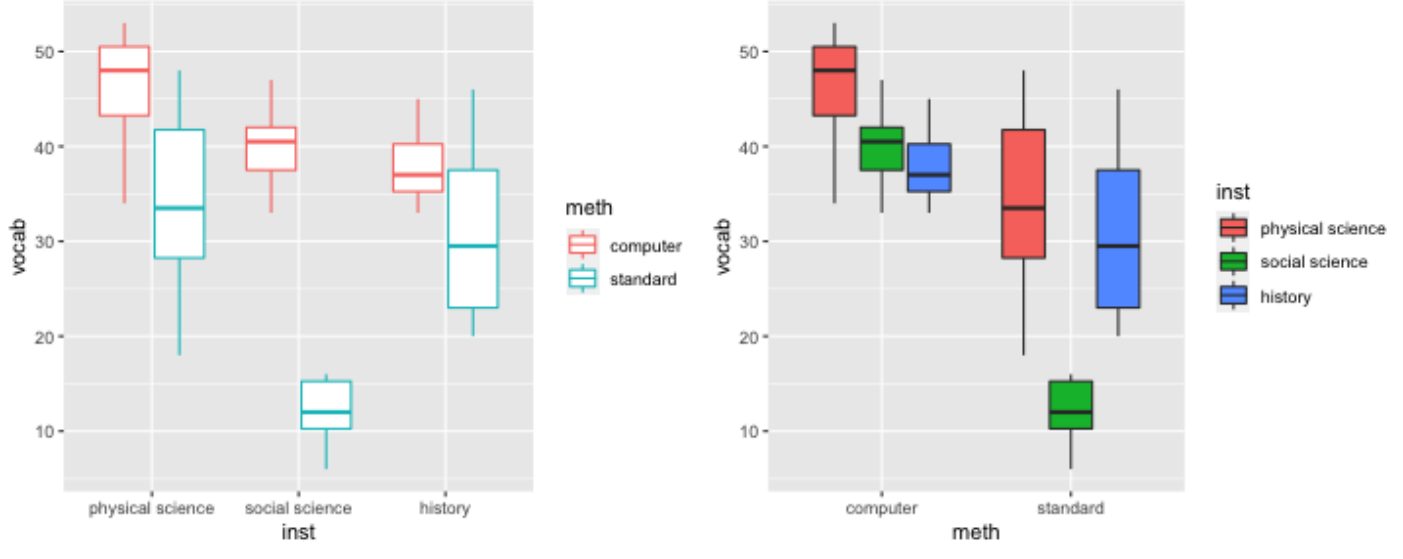
```
inst_plot <- ggplot(data, aes(x = inst, y = vocab)) +
  geom_boxplot()
meth_plot <- ggplot(data, aes(x = meth, y = vocab)) +
  geom_boxplot()
gridExtra::grid.arrange(inst_plot, meth_plot, nrow = 1) ## nrow =
```



Clustered Boxplots

Same process but you'll need to specify the other IV using `color =`, or `fill =`.

```
inst_color <- ggplot(data, aes(x = inst, y = vocab, color = meth)
  geom_boxplot()
meth_fill <- ggplot(data, aes(x = meth, y = vocab, fill = inst))
  geom_boxplot()
gridExtra::grid.arrange(inst_color, meth_fill, nrow = 1)
```



Two-Way ANOVA

```
needs(rstatix, emmeans)
```

Two–Way ANOVA

For a two–way ANOVA we just add our additional IV to the right–side of the formula. We specify the interaction using `inst*meth`.

```
m1 <- anova_test(data, formula = vocab ~ inst + meth + inst:meth
                 detailed = TRUE, type = 3, effect.size = "pes")
```

```
## Coefficient covariances computed by hccm()
```

```
m1
```

```
## ANOVA Table (type III tests)
##
##        Effect   SSn  SSd DFn DFd       F        p p<.05   pes
## 1 (Intercept) 40401 1668   1  30 726.637 1.39e-22     * 0.960
## 2        inst  1194 1668   2  30  10.737 3.04e-04     * 0.417
## 3        meth  2209 1668   1  30  39.730 5.99e-07     * 0.570
## 4   inst:meth   722 1668   2  30   6.493 5.00e-03     * 0.302
```

Levene's Test

```
car::leveneTest(vocab ~ inst*meth, data = data, center = "mean")
```

```
## Levene's Test for Homogeneity of Variance (center = "mean")
##       Df F value  Pr(>F)
## group  5  2.0579 0.09877 .
##       30
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Marginal Means

To get estimated marginal means for an interaction, we need to group by one of the IVs and then run our `emmeans_test` with the other. This just gives us the mean differences and significance tests.

```
means <- data %>%
  group_by(meth) %>%
 emmeans_test(vocab ~ inst, p.adjust.method = "holm", detailed =
paged_table(means)
```

	meth <chr>	term <chr>	.y. <chr>	group1 <chr>
1	computer	inst	vocab	physical science
2	computer	inst	vocab	physical science
3	computer	inst	vocab	social science
4	standard	inst	vocab	physical science

Marginal Means

To get the actual marginal means, we just run `get_emmeans()` on our `means` object.

```
paged_table(
  get_emmeans(means)
)
```

meth <fct>	inst <fct>	emmean <dbl>	se <dbl>	df <dbl>
computer	physical science	46	3.04412	30
computer	social science	40	3.04412	30
computer	history	38	3.04412	30
standard	physical science	34	3.04412	30
standard	social science	12	3.04412	30
standard	history	31	3.04412	30

6 rows | 1–5 of 8 columns

Univariate Comparisons

Since we have an interaction, we are going to check the significance of method (`meth`) on each level of instruction (`inst`). We are telling it to use SS and df from our full model with `error = model`.

```
model <- lm(vocab ~ inst + meth + inst:meth, data = data)

data %>%
  group_by(inst) %>%
  anova_test(vocab ~ meth, error = model)
```

```
## Coefficient covariances computed by hccm()
## Coefficient covariances computed by hccm()
## Coefficient covariances computed by hccm()
```

	inst <fct>	Effect <chr>	DFn <dbl>	DFd <dbl>
1	physical science	meth	1	30
2	social science	meth	1	30
3	history	meth	1	30

Univariate Comparisons

We can instead test the effect of *instruction* on each level of *method* by switching the two IVs.

```
data %>%
  group_by(meth) %>%
  anova_test(vocab ~ inst, error = model)
```

```
## Coefficient covariances computed by hccm()
## Coefficient covariances computed by hccm()

## # A tibble: 2 x 8
##   meth     Effect   DFn   DFd     F          p `p<.05`   ges
## * <fct>    <chr>  <dbl> <dbl> <dbl>      <dbl> <chr>   <dbl>
## 1 computer inst       2    30  1.87 0.172      ""      0.111
## 2 standard inst       2    30 15.4  0.0000255 "*"     0.506
```

Interaction Plots

This the most straightforward way to produce an interaction plot. For those interested, I'll add a slide on how to extract means for use in `ggplot` by early next week.

```
with(data, {
interaction.plot(x.factor = inst,
                 trace.factor = meth,
                 response = vocab,
                 type = "l")
})
```

Final Notes

The tough thing about running this in R is having to ask for everything and then having lots of separate output to review. For now it'll be good to know how to ask for something specific and not rely on something too automated. However, I'll look into finding or writing some custom functions that help tidy up all this output.