EDUC 640

Two-Way Factorial ANOVA

Chris Ives

Contents

- Plotting Means
- Recoding/Reordering Factors
- Contrasts using `emmeans`

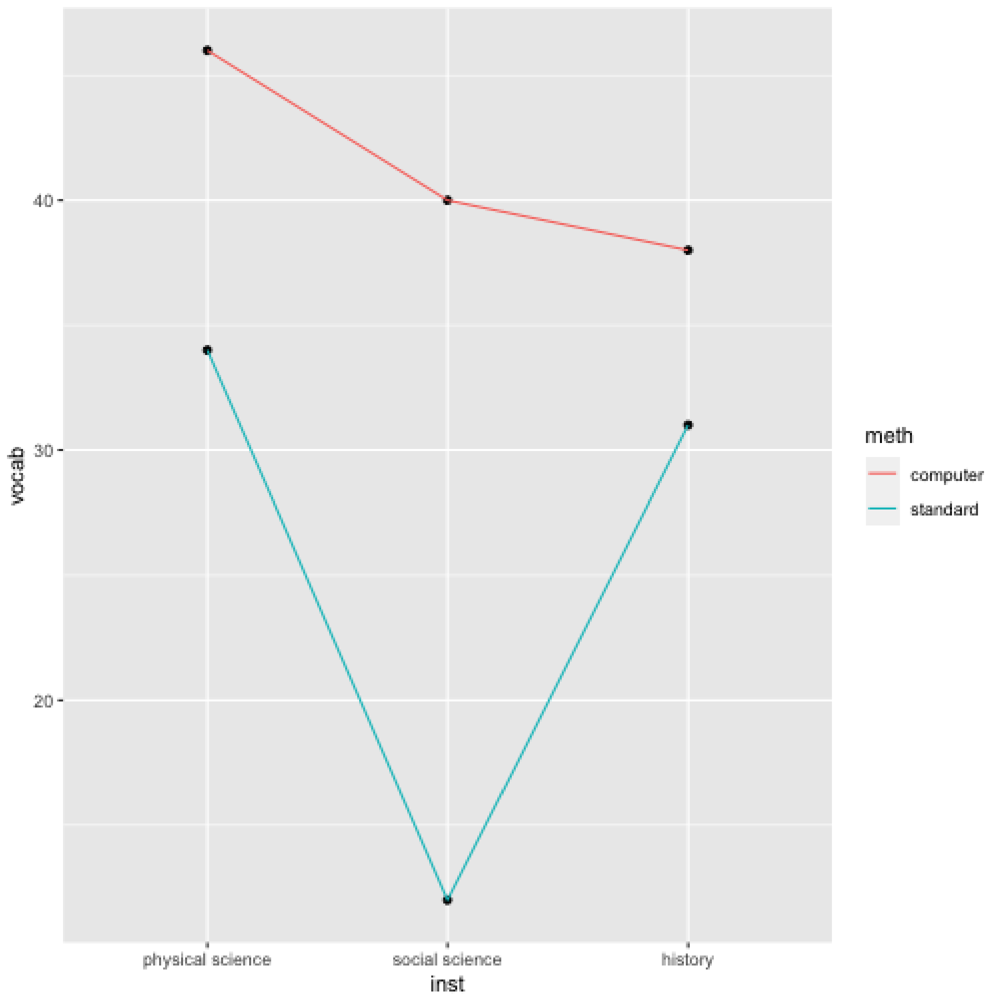
Plot Marginal Means

Here I am creating a new summary data frame. After grouping by `meth` and `inst`, I just make `vocab` equal to the mean of those groups. This dataframe is another way of getting marginal mean values.

```
means <- lb5 %>%
  group_by(inst, meth) %>%
  summarise(vocab = mean(vocab))
```

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

```r
ggplot(means, aes(x = inst, y = vocab, group = meth)) +
  geom_point() +
  geom_line(aes(color = meth))
```



Contrasts using `emmeans`

To do two–factor contrasts we can use the `emmeans` package. You will probably find this easier than the previous approaches where I had you specify your coding in a matrix. Whether you prefer this for one–way contrasts will be up to you.

Two Factor Groupings

First we run our model specification using **lm**. Then we run **emmeans** on that object to calculate the marginal means. There are multiple ways to use **emmeans** if you plan to run follow–up pairwise comparisons. This first way will run comparisons within the **by** grouping.

```
model <- lm(vocab ~ meth + inst + meth:inst, lb5)
means_1 <- emmeans(model, "inst", by = "meth")
means_1
```

```
## meth = computer:
##  inst             emmean   SE df lower.CL upper.CL
##  physical science     46 3.04 30    39.78     52.2
##  social science       40 3.04 30    33.78     46.2
##  history              38 3.04 30    31.78     44.2
##
## meth = standard:
##  inst             emmean   SE df lower.CL upper.CL
##  physical science     34 3.04 30    27.78     40.2
##  social science       12 3.04 30     5.78     18.2
##  history              31 3.04 30    24.78     37.2
##
```

```r
pairs(means_1)
```

```
## meth = computer:
##  contrast                            estimate   SE df t.ratio p.value
##  physical science - social science          6 4.31 30   1.394  0.3568
##  physical science - history                 8 4.31 30   1.858  0.1684
##  social science - history                   2 4.31 30   0.465  0.8883
##
## meth = standard:
##  contrast                            estimate   SE df t.ratio p.value
##  physical science - social science         22 4.31 30   5.110  <.0001
##  physical science - history                 3 4.31 30   0.697  0.7671
##  social science - history                 -19 4.31 30  -4.413  0.0003
##
## P value adjustment: tukey method for comparing a family of 3 estimates
```

This way doesn't "nest" one factor in the other and will calculate all possible comparisons.

```
means_2 <- emmeans(model, ~inst*meth)


means_2
```

```
##  inst             meth     emmean   SE df lower.CL upper.CL
##  physical science computer     46 3.04 30    39.78     52.2
##  social science   computer     40 3.04 30    33.78     46.2
##  history          computer     38 3.04 30    31.78     44.2
##  physical science standard     34 3.04 30    27.78     40.2
##  social science   standard     12 3.04 30     5.78     18.2
##  history          standard     31 3.04 30    24.78     37.2
##
## Confidence level used: 0.95
```

```
pairs(means_2)
```

contrast <chr>
physical science computer – social science computer
physical science computer – history computer
physical science computer – physical science standard
physical science computer – social science standard
physical science computer – history standard
social science computer – history computer
social science computer – physical science standard
social science computer – social science standard
social science computer – history standard
history computer – physical science standard

1–10 of 15 rows | 1–1 of 6 columns   Previous 1 2 Next

Specifying contrasts

We'll be using the listing structure from the second method.

We just have to specify our contrasts into a list.

```
means_2
```

```
##  inst             meth     emmean   SE df lower.CL upper.CL
##  physical science computer     46 3.04 30    39.78     52.2
##  social science   computer     40 3.04 30    33.78     46.2
##  history          computer     38 3.04 30    31.78     44.2
##  physical science standard     34 3.04 30    27.78     40.2
##  social science   standard     12 3.04 30     5.78     18.2
##  history          standard     31 3.04 30    24.78     37.2
##
## Confidence level used: 0.95
```

```
contrasts <- list(
  hyp1 = c(1, -.5, -.5, -1, .5, .5),
  hyp2 = c(1, -1, 0, -1, 1, 0)
)
```

Much easier!!

```r
contrast(means_2, contrasts, adjust = "holm")
```

```
##  contrast estimate   SE df t.ratio p.value
##  hyp1         -5.5 5.27 30 -1.043  0.3052
##  hyp2        -16.0 6.09 30 -2.628  0.0268
##
## P value adjustment: holm method for 2 tests
```

Effect Sizes

If you want partial eta effect sizes (like SPSS output), you'll need to calculate them from the contrast output using the function below.

```
effectsize::t_to_eta2(
  t = c(-1.043, -2.628),
  df_error = 30
)
```

```
## Eta2 (partial) |       90% CI
## -----------------------------
## 0.03           | [0.00, 0.19]
## 0.19           | [0.02, 0.39]
```

How would you do this for a One–way ANOVA? I'll demonstrate the process from the beginning.

```r
m1 <- lm(vocab ~ inst, lb5)

means <- emmeans(m1, ~inst)
means
```

```
##  inst               emmean   SE df lower.CL upper.CL
##  physical science   40.0 3.41 33     33.1     46.9
##  social science     26.0 3.41 33     19.1     32.9
##  history            34.5 3.41 33     27.6     41.4
##
## Confidence level used: 0.95
```

Specify contrast

```
contrasts <- list(
  hyp1 = c(1, -.5, -.5),
  hyp2 = c(1, 0, -1)
  )

contrast(means, contrasts)
```

```
##  contrast estimate   SE df t.ratio p.value
##  hyp1         9.75 4.17 33 2.336   0.0257
##  hyp2         5.50 4.82 33 1.141   0.2620
```

Calculate partial eta effect size using previous **t-value** from previous output.

```
effectsize::t_to_eta2(
  t = c(2.336),
  df_error = 33
)
```

```
## Eta2 (partial) |       90% CI
## -----------------------------
## 0.14           | [0.01, 0.33]
```

Appendix

All of the following slides reflect less efficient ways of doing contrasts. I am just leaving them in if you want to see some examples of renaming levels and uniting variables.

Recoding Data

We have to start by combining our two factors into one factor with six levels. The level names are a little long for my taste so here I am shortening them. First check the orders of the levels so your names are assigned appropriately.

```
levels(lb5$meth)
```

```
## [1] "computer" "standard"
```

```
levels(lb5$meth) <- c("comp", "stan")
```

```
levels(lb5$inst)
```

```
## [1] "physical science" "social science"   "history"
```

```
levels(lb5$inst) <- c("phys", "soc", "hist")
```

Next I join the two factors into one with `unite`. `col =` specifies the name of the new column, followed by the columns I am joining. I chose to separate them with "_" and set `remove = FALSE` so I don't delete the old variables.

```
lb5 <- lb5 %>%
  unite(col = ivs, meth, inst, sep = "_", remove = FALSE)

head(lb5)
```

```
##   idnum vocab       ivs inst meth
## 1     1    53 comp_phys phys comp
## 2     2    49 comp_phys phys comp
## 3     3    47 comp_phys phys comp
## 4     4    42 comp_phys phys comp
## 5     5    51 comp_phys phys comp
## 6     6    34 comp_phys phys comp
```

Last bit of data prep is order them in a way that will make sense for me when I code out contrasts later. This order reflects what's in Gina's slides.

```
lb5$ivs <- ordered(lb5$ivs, c("comp_phys", "comp_soc", "comp_his
levels(lb5$ivs)
```

```
## [1] "comp_phys" "comp_soc"  "comp_hist" "stan_phys" "stan_soc"  "stan_hi
```

Emmeans

Start by specifying your model and then running emmeans on that model. Note that I am using `ivs`, which is our combined two factor variable.

```
m1 <- lm(vocab ~ ivs, data = lb5)
emm <- emmeans(m1, ~ ivs)
```

Then we will check the level order and assign a vectors to each. We have 6 levels so the vector is 6 numbers long. A `1` in the vector means I am saving the mean of that level to my object. So, the mean of "comp_phys" is saved to `A1B1`.

```
levels(lb5$ivs)
```

```
## [1] "comp_phys" "comp_soc"  "comp_hist" "stan_phys" "stan_soc"  "stan_hi
```

```
A1B1 <- c(1, 0, 0, 0, 0, 0)
A1B2 <- c(0, 1, 0, 0, 0, 0)
A1B3 <- c(0, 0, 1, 0, 0, 0)
A2B1 <- c(0, 0, 0, 1, 0, 0)
A2B2 <- c(0, 0, 0, 0, 1, 0)
A2B3 <- c(0, 0, 0, 0, 0, 1)
```

Hypothesis 1

Now we can specify our contrasts (reference slides 445–446 in Lab 5). This runs the contrasts so it prints the coding scheme.

```
contrast(emm, method = list(
  (A1B1 - (A1B2 + A1B3)/2) -
  (A2B1 - (A2B2 + A2B3)/2)
  ))
```

```
##  contrast                                estimate   SE df t.ratio p.value
##  c(1, -0.5, -0.5, -1, 0.5, 0.5)             -5.5 5.27 30 -1.043  0.3052
```

Here I name the contrast.

```
contrast(emm, method = list(
  "Hyp1" = (A1B1 - (A1B2 + A1B3)/2) -
  (A2B1 - (A2B2 + A2B3)/2)
))
```

```
##  contrast estimate   SE df t.ratio p.value
```

Hypothesis 2

Same process for our second contrast.

```
contrast(emm, method = list(
  (A1B1 - A1B2) - (A2B1 - A2B2)
  ))
```

```
##  contrast                     estimate   SE df t.ratio p.value
##  c(1, -1, 0, -1, 1, 0)             -16 6.09 30 -2.628  0.0134
```

```
contrast(emm, method = list(
  "Hyp2" = (A1B1 - A1B2) - (A2B1 - A2B2)
  ))
```

```
##  contrast estimate   SE df t.ratio p.value
##  Hyp2          -16 6.09 30 -2.628  0.0134
```

Combining Contrasts

If I was planning to put them into a markdown document, I'd probably want to write it all out with one command and output.

```
contrast(emm, method = list(
  "Hyp1" = (A1B1 - (A1B2 + A1B3)/2) -
  (A2B1 - (A2B2 + A2B3)/2),
  "Hyp2" = (A1B1 - A1B2) - (A2B1 - A2B2)
))
```

```
##  contrast estimate   SE df t.ratio p.value
##  Hyp1         -5.5 5.27 30 -1.043  0.3052
##  Hyp2        -16.0 6.09 30 -2.628  0.0134
```

Here's the process using the method I described in the Appendix of Wk1–3 slides.

```
contrast0 <- c(1, -.5, -.5, -1, .5, .5)
mat.temp <- rbind(constant = 1/6, contrast0)
mat.temp
```

```
##                 [,1]       [,2]       [,3]       [,4]      [,5]      [,6]
## constant   0.1666667  0.1666667  0.1666667  0.1666667 0.1666667 0.166666
## contrast0 1.0000000 -0.5000000 -0.5000000 -1.0000000 0.5000000 0.500000
```

```
mat <- MASS::ginv(mat.temp)
mat <- mat[ , -1]
mat
```

```
## [1]  0.3333333 -0.1666667 -0.1666667 -0.3333333  0.1666667  0.1666667
```

Remember I only specified one contrast so we only pay attention to the first coefficient (that's not the intercept).

```
m_contrasts <- lm(vocab ~ ivs, data=lb5, contrasts = list(ivs = r
summary(m_contrasts)
```

```
##
## Call:
## lm(formula = vocab ~ ivs, data = lb5, contrasts = list(ivs = mat))
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -16.00  -3.25  -0.50   4.00  15.00
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   33.500      1.243  26.956   <2e-16 ***
## ivs1          -5.500      5.273  -1.043   0.3052
## ivs2          -1.019      3.044  -0.335   0.7401
## ivs3          -6.219      3.044  -2.043   0.0499 *
## ivs4         -24.620      3.044  -8.088    5e-09 ***
## ivs5          -5.620      3.044  -1.846   0.0748 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.457 on 30 degrees of freedom
## Multiple R-squared:  0.7121,    Adjusted R-squared:  0.6641
## F-statistic: 14.84 on 5 and 30 DF,  p-value: 2.382e-07
```