PS6

Sam Olson

Outline

- Q1: Draft
- Q2: Edited, G2G
- Q3: Edited, G2G

Q1

Consider the dataset `pigs` provided in the R package `emmeans`. The data can be accessed in R with the following commands.

```r
library(emmeans)
names(pigs)
```

```
## [1] "source"  "percent" "conc"
```

To learn a more about the data, type `?pigs` at the R prompt. For the purposes of this problem, use the natural logarithm of the variable `conc` as the response. Consider both `source` and `percent` as categorical factors. Assume the cell-means model with one unrestricted treatment mean for each combination of `source` and `percent`.

```r
lnConc <- log(pigs$conc)
pigs$percent <- factor(pigs$percent)
class(pigs$percent)
```

```
## [1] "factor"
```

a)

Generate an ANOVA table with Type I (sequential) sums of squares for `source`, `percent`, `source × percent`, `error`, and `corrected total`. In addition to sums of squares, your ANOVA table should include degrees of freedom, mean squares, F statistics, and p-values where appropriate.

Type 1 Sums of Squares are the default given when using the `anova` function in R. Hence:

```
baseDat <- lm(lnConc ~ source + percent + source*percent, data=pigs)
type1Dat <- anova(baseDat)
type1_df <- data.frame(type1Dat)

total <- c(sum(type1_df$`Df`), sum(type1_df$`Sum.Sq`), NA, NA, NA)
type1_df <- rbind(type1_df, total)
rownames(type1_df)[nrow(type1_df)] <- "Total"

print(type1_df, digits = 6)
```

```
##                Df    Sum.Sq   Mean.Sq   F.value      Pr..F.
## source          2 0.6301386 0.3150693 23.311284 1.34314e-05
## percent         3 0.3173579 0.1057860  7.826871 1.69937e-03
## source:percent  6 0.0750848 0.0125141  0.925893 5.01099e-01
## Residuals      17 0.2297676 0.0135157        NA          NA
## Total          28 1.2523490        NA        NA          NA
```

b)

Generate an ANOVA table with Type II sums of squares for `source`, `percent`, `source × percent`, `error`, and `corrected total`. In addition to sums of squares, your ANOVA table should include degrees of freedom, mean squares, F statistics, and p-values where appropriate.

The `car` package allows more flexible usage of Types of Suns of Squares used for anova. Hence:

```
library(car)
type2Dat <- car::Anova(baseDat, type = 2)

type2_df <- as.data.frame(type2Dat)

# Compute Mean Square (MS) = SS / df
type2_df$MeanSq <- type2_df$`Sum Sq` / type2_df$Df

# Reorder columns to match anova() output
type2_df <- type2_df[, c("Df", "Sum Sq", "MeanSq", "F value", "Pr(>F)")]

total <- c(sum(type2_df$`Df`), sum(type2_df$`Sum Sq`), NA, NA, NA)
type2_df <- rbind(type2_df, total)
rownames(type2_df)[nrow(type2_df)] <- "Total"
# Print with higher precision
print(type2_df, digits = 6)
```

```
##                Df    Sum Sq    MeanSq   F value      Pr(>F)
## source          2 0.7647592 0.3823796 28.291424 3.90135e-06
## percent         3 0.3173579 0.1057860  7.826871 1.69937e-03
## source:percent  6 0.0750848 0.0125141  0.925893 5.01099e-01
## Residuals      17 0.2297676 0.0135157        NA          NA
## Total          28 1.3869696        NA        NA          NA
```

c)

Generate an ANOVA table with Type III sums of squares for `source`, `percent`, `source × percent`, `error`, and `corrected total`. In addition to sums of squares, your ANOVA table should include degrees of freedom, mean squares, F statistics, and p-values where appropriate.

Quick note on the continued use of car::Anova for this problem: By default R uses the baseline constraint when making these calculations, and in R the baseline is the combination of the first factor levels. So effectively we just need to change the type of contrast being used when calculating the Anova table from treatment aka `contr.treatment` to sums aka `contra.sum` (which corresponds to a sum-to-zero constraint) and proceed, bearing in mind Type 3 Sums of Squares are 'marginal' Sums of Squares, hence the reason for this adjustment.

```
options(digits = 6)

# car Anova function needs some adjustment for Type 3
contrasts(pigs$source) <- contr.sum
contrasts(pigs$percent) <- contr.sum

type3Dat <- car::Anova(baseDat, type = 3)
type3_df <- as.data.frame(type3Dat)

# Add MSE
type3_df$MeanSq <- type3_df$`Sum Sq` / type2_df$Df
type3_df <- type3_df[, c("Df", "Sum Sq", "MeanSq", "F value", "Pr(>F)")]

type3_df <- type3_df[-1, ]

total <- c(sum(type3_df$`Df`), sum(type3_df$`Sum Sq`), NA, NA, NA)
type3_df <- rbind(type3_df, total)
rownames(type3_df)[nrow(type3_df)] <- "Total"

print(type3_df, digits = 6)
```

```
##                Df    Sum Sq     MeanSq  F value    Pr(>F)
## source          2 0.1395725 0.04652418 5.163332 0.0176941
## percent         3 0.0694365 0.01157275 1.712485 0.2022617
## source:percent  6 0.0750848 0.00441675 0.925893 0.5010985
## Residuals      17 0.2297676 0.00820599       NA        NA
## Total          28 0.5138615         NA       NA        NA
```

d)

Find LSMeans for `source` and `percent`.

```
library(tidyverse)

summary_table <- pigs |>
  group_by(source, percent) |>
  mutate(count = n())

print(summary_table, digits = 6)
```

```
## # A tibble: 29 x 4
## # Groups:   source, percent [12]
##    source percent  conc count
##    <fct>  <fct>   <dbl> <int>
##  1 fish   9        27.8     2
##  2 fish   9        23.7     2
##  3 fish   12       31.5     3
##  4 fish   12       28.5     3
##  5 fish   12       32.8     3
##  6 fish   15       34       2
##  7 fish   15       28.3     2
##  8 fish   18       30.6     3
##  9 fish   18       32.7     3
## 10 fish   18       33.7     3
## # i 19 more rows
```

Don't have balance, so need to explicitly calculate LSmeans (they are not the same as OLS).

```
library(tidyverse)
library(emmeans)

# Fit the model
modDat <- lm(lnConc ~ source * percent, data = pigs)

# Compute LS Means
lsmeans_table <- emmeans(modDat, ~ source * percent)

# Convert to a data frame
lsmeans_df <- as.data.frame(lsmeans_table)

# Reshape to wide format
lsmeans_wide <- lsmeans_df |>
  select(source, percent, emmean) |>
  pivot_wider(names_from = percent, values_from = emmean)

lsmeans_wide
```

```
## # A tibble: 3 x 5
##   source   `9`  `12`  `15`  `18`
##   <fct>  <dbl> <dbl> <dbl> <dbl>
## 1 fish    3.25  3.43  3.43  3.48
```

```
## 2 soy     3.54  3.68  3.67  3.76
## 3 skim    3.56  3.76  3.90  4.09
```

```
lsmeans_wide <- lsmeans_wide |>
  mutate(Row_Avg = rowMeans(select(lsmeans_wide, -source), na.rm = TRUE))

lsmeans_wide[-c(2:5)]
```

```
## # A tibble: 3 x 2
##   source Row_Avg
##   <fct>    <dbl>
## 1 fish      3.40
## 2 soy       3.66
## 3 skim      3.83
```

```
# Compute column-wise averages (across sources)
col_avg <- lsmeans_wide |>
  summarise(across(-source, mean, na.rm = TRUE)) |>
  mutate(source = "Column_Avg")

col_avg
```

```
## # A tibble: 1 x 6
##     `9`  `12`  `15`  `18` Row_Avg source
##   <dbl> <dbl> <dbl> <dbl>   <dbl> <chr>
## 1  3.45  3.62  3.67  3.78    3.63 Column_Avg
```

e)

Consider simplifying the model so that `percent` is treated like a quantitative variable with linear effects on `log(conc)` and linear interactions; i.e.,

```
lm(y ~ source + percent + source:percent)
```

where `y=log(conc)` and `percent` is numeric. Does such a model fit adequately relative to the cell-means model? Conduct a lack of fit test and report the results.

Back to the land of vanilla `anova`. We can just put the two models in and compare directly.

```
# Reduced model: Treat percent as numeric
reduced <- lm(lnConc ~ source + as.numeric(percent) + source:as.numeric(percent), data = pigs)

# Full model: Treat percent as a categorical factor
full <- lm(lnConc ~ source * factor(percent), data = pigs)

# Compare the two models with ANOVA
anova(reduced, full)
```

```
## Analysis of Variance Table
##
## Model 1: lnConc ~ source + as.numeric(percent) + source:as.numeric(percent)
## Model 2: lnConc ~ source * factor(percent)
##   Res.Df    RSS Df Sum of Sq     F Pr(>F)
## 1     23 0.2629
## 2     17 0.2298  6   0.03314 0.409  0.863
```

We have evidence to support the reduced model being adequate in lieu of the more complex, cell means model.

f)

The reduced model fit in part (e) implies that, for each `source`, there is a linear relationship between the expected log concentration and percentage. Based on the fit of the reduced model in part (e), provide the estimated linear relationship for each `source`.

$$y_{ijk} = \mu + \alpha_i + \beta x_{ij} + \gamma_i x_{ij} + \epsilon_{ijk}$$

Based on this model, the estimated linear relationship for each source is as follows.

Fish

$$(\hat{\mu} + \hat{\alpha}_1) + (\hat{\beta} + \hat{\gamma}_1) \cdot x_{1j} = 3.1164 + 0.0211 x_{1j}$$

Soy

$$(\hat{\mu} + \hat{\alpha}_2) + (\hat{\beta} + \hat{\gamma}_2) \cdot x_{2j} = (3.1164 + 0.2517) + (0.0211 + 0.0006) x_{2j} = 3.3681 + 0.0217 x_{2j}$$

Skim

$$(\hat{\mu} + \hat{\alpha}_3) + (\hat{\beta} + \hat{\gamma}_3) \cdot x_{3j} = (3.1164 - 0.0672) + (0.0211 + 0.0369) x_{3j} = 3.0492 + 0.058 x_{3j}$$

Q2

Consider the plant density example discussed in slide set 6.

Add image of Slide here:

a)

For each of the tests in the ANOVA table on slide 38, provide a vector c so that a test of

$$H_0 : c^T\beta = 0$$

would yield the same statistic and p-value as the ANOVA test. (You can use R to help you with the computations like we did on slides 45 and 46 of slide set 6.) Label these vectors c_1, c_2, c_3, and c_4 for the linear, quadratic, cubic, and quartic tests, respectively.

```r
proj <- function(x) {
  x %*% MASS::ginv(t(x) %*% x) %*% t(x)
}

# Had to dig through Canvas to find the supplemental R handout to find this. Smh
# Or, I guess not, there's a lot there to borrow for this problem
d <- read.delim("https://dnett.github.io/S510/PlantDensity.txt")
names(d) <- c("x","y")

n <- nrow(d)
x <- (d$x-mean(d$x))/10
# Iteratively, we just need to add a higher order term to the prior design matrix
x1 <- matrix(1, nrow = n, ncol = 1)
x2 <- cbind(x1,x)
x3 <- cbind(x2,x^2)
x4 <- cbind(x3,x^3)
x5 <- matrix(model.matrix(~0+factor(x)),nrow <- n)
```

```r
p1 <- proj(x1)
p2 <- proj(x2)
p3 <- proj(x3)
p4 <- proj(x4)
p5 <- proj(x5)

((p2-p1)%*%x5)[1,] *5 ## linear
```

```
## [1]  2  1  0 -1 -2
```

```r
((p3-p2)%*%x5)[1,] *7 ## quadratic
```

```
## [1]  2 -1 -2 -1  2
```

```r
((p4-p3)%*%x5)[1,] *10 ## cubic
```

```
## [1]  1 -2  0  2 -1
```

```r
((p5-p4)%*%x5)[1,] *70 ## quartic
```

```
## [1]  1 -4  6 -4  1
```

From the above output, we have:

$$\mathbf{c}_1^\top = [2, 1, 0, -1, -2]\mathbf{c}_2^\top = [2, -1, -2, -1, 2]\mathbf{c}_3^\top = [1, -2, 0, 2, -1]\mathbf{c}_4^\top = [1, -4, 6, -4, 1]$$

b)

Are $c_1^T\beta$, $c_2^T\beta$, $c_3^T\beta$, and $c_4^T\beta$ contrasts? Explain.

All $\mathbf{c}_i^\top\boldsymbol{\beta}$ are contrasts because:

$$\mathbf{c}_i^\top\mathbf{1} = 0 \quad \text{for } i = 1, 2, 3, 4$$

Explicitly:

$$\mathbf{c}_1^\top\mathbf{1} = 2 + 1 + 0 - 1 - 2 = 0$$

$$\mathbf{c}_2^\top\mathbf{1} = 2 - 1 - 2 - 1 + 2 = 0$$

$$\mathbf{c}_3^\top\mathbf{1} = 1 - 2 + 0 + 2 - 1 = 0$$

$$\mathbf{c}_4^\top\mathbf{1} = 1 - 4 + 6 - 4 + 1 = 0$$

And not only are these contrasts, but they are orthogonal contrasts because they satisfy:

$$\mathbf{c}_i^\top\mathbf{c}_j = 0, \quad \text{for } i \neq j$$

c)

Are $c_1^T\beta$, $c_2^T\beta$, $c_3^T\beta$, and $c_4^T\beta$ orthogonal? Explain.

Preempted this question a bit at the end of part b), but yes, these contrasts are orthogonal, and here is more detail on why beyond the expression they satisfy.

So, the initial expression to check is:

$$\mathbf{c}_i^\top \mathbf{c}_j = 0, \quad \text{for } i \neq j$$

Explicitly:

4 choose 2 cases to check, 6 total cases:

$$\mathbf{c}_1^\top \mathbf{c}_2 = (2)(2) + (1)(-1) + (0)(-2) + (-1)(-1) + (-2)(2) = 4 - 1 + 0 + 1 - 4 = 0$$

$$\mathbf{c}_1^\top \mathbf{c}_3 = (2)(1) + (1)(-2) + (0)(0) + (-1)(2) + (-2)(-1) = 2 - 2 + 0 - 2 + 2 = 0$$

$$\mathbf{c}_1^\top \mathbf{c}_4 = (2)(1) + (1)(-4) + (0)(6) + (-1)(-4) + (-2)(1) = 2 - 4 + 0 + 4 - 2 = 0$$

$$\mathbf{c}_2^\top \mathbf{c}_3 = (2)(1) + (-1)(-2) + (-2)(0) + (-1)(2) + (2)(-1) = 2 + 2 + 0 - 2 - 2 = 0$$

$$\mathbf{c}_2^\top \mathbf{c}_4 = (2)(1) + (-1)(-4) + (-2)(6) + (-1)(-4) + (2)(1) = 2 + 4 - 12 + 4 + 2 = 0$$

$$\mathbf{c}_3^\top \mathbf{c}_4 = (1)(1) + (-2)(-4) + (0)(6) + (2)(-4) + (-1)(1) = 1 + 8 + 0 - 8 - 1 = 0$$

Note: The above is a simplification that works for the example given because of the design matrix used in this problem.

The "base" definition we reference for testing whether something is orthogonal is: Any two estimable linear combinations $c_i^T\boldsymbol{\beta}$ and $c_j^T\boldsymbol{\beta}$ are orthogonal if and only if:

$$\mathbf{c}_i^T(\mathbf{X^T X})^{-}\mathbf{c}_j = 0 \quad \text{for } i \neq j$$

We use the full design matrix (`x5` from above) for the calculations. Since $\mathbf{X^T X}$ is full rank, we can compute its inverse directly rather than using a generalized inverse. Specifically, because $\mathbf{X^T X}$ is a scalar multiple of the identity matrix, we ensure it is invertible, meaning its inverse is unique and can be computed using the `solve` function in R.

Evaluating the above, we get the following matrices:

```r
require(MASS)
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
x5
```

```
##       [,1] [,2] [,3] [,4] [,5]
##  [1,]    1    0    0    0    0
##  [2,]    1    0    0    0    0
##  [3,]    1    0    0    0    0
##  [4,]    0    1    0    0    0
##  [5,]    0    1    0    0    0
##  [6,]    0    1    0    0    0
##  [7,]    0    0    1    0    0
##  [8,]    0    0    1    0    0
##  [9,]    0    0    1    0    0
## [10,]    0    0    0    1    0
## [11,]    0    0    0    1    0
## [12,]    0    0    0    1    0
## [13,]    0    0    0    0    1
## [14,]    0    0    0    0    1
## [15,]    0    0    0    0    1
```

```r
t(x5) %*% x5
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    3    0    0    0    0
## [2,]    0    3    0    0    0
## [3,]    0    0    3    0    0
## [4,]    0    0    0    3    0
## [5,]    0    0    0    0    3
```

```r
fractions(solve(t(x5) %*% x5))
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,] 1/3    0    0    0    0
## [2,]   0  1/3    0    0    0
## [3,]   0    0  1/3    0    0
## [4,]   0    0    0  1/3    0
## [5,]   0    0    0    0  1/3
```

Thus, in this case,

$$\mathbf{c}_i^T(\mathbf{X^TX})^{-1}\mathbf{c}_j = \mathbf{c}_i^T\mathbf{c}_j/3$$

so that linear combinations $\mathbf{c}_i^T\boldsymbol{\beta}$ and $\mathbf{c}_j^T\boldsymbol{\beta}$ are orthogonal if and only if $\mathbf{c}_i^T\mathbf{c}_j = 0$. (Multiplying each side by 3 for simplicity).

The rest of the results (that our contrasts are orthogonal) follow from the explicit derivations.

Q3

Suppose $\mathbf{H}$ is a symmetric matrix. Prove that H is nonnegative definite if and only if all its eigenvalues are nonnegative. (If you wish, you may use the Spectral Decomposition Theorem in your proof.)

I do want to use Spectral Decomposition, thank you!

Spectral Decomposition Theorem:

For $\mathbf{H}$ is a symmetric matrix, then:

$$\mathbf{H} = \mathbf{P}\Lambda\mathbf{P}^\top = \sum_{i=1}^{n} \lambda_i \mathbf{p}_i \mathbf{p}_i^\top$$

where $\mathbf{p}_i$ are orthonormal eigenvectors of $\mathbf{H}$.

The general approach for solving an iff proof is to prove both directions hold. To that end:

Direction 1

Assume we have $\mathbf{H}$, a symmetric, nonnegative definite matrix.

By definition, $\mathbf{H}$ being nonnegative definite, means the following holds:

$$\mathbf{p}_i^\top \mathbf{H} \mathbf{p}_i \geq 0$$

for any $\mathbf{p}_i$, where $i = 1, \ldots, n$.

Unfurling $\mathbf{H}$, we may rewrite:

$$\mathbf{p}_i^\top \mathbf{H} \mathbf{p}_i = \mathbf{p}_i^\top \left(\sum_{j=1}^{n} \lambda_j \mathbf{p}_j \mathbf{p}_j^\top \right) \mathbf{p}_i = \sum_{j=1}^{n} \lambda_j \mathbf{p}_i^\top \mathbf{p}_j \mathbf{p}_j^\top \mathbf{p}_i$$

By Special Decomposition, we know $\mathbf{p}_i$ are orthonormal, meaning:

$$\mathbf{p}_i^\top \mathbf{p}_j = 0 \quad \text{for all } i \neq j, \quad \text{and} \quad \mathbf{p}_i^\top \mathbf{p}_i = 1$$

Allowing us to further simplify:

$$\mathbf{p}_i^\top \mathbf{H} \mathbf{p}_i = \lambda_i \mathbf{p}_i^\top \mathbf{p}_i \mathbf{p}_i^\top \mathbf{p}_i = \lambda_i$$

And because $\mathbf{H}$ is nonnegative definite, it then follows that:

$$\lambda_i \geq 0, \quad \forall i = 1, \ldots, n$$

Direction one complete!

Direction 2

We start by assuming that for a symmetric matrix $\mathbf{H}$, all its eigenvalues are nonnegative.

Given our assumption, we may restate as $\lambda_i \geq 0$ for $i = 1, \ldots, n$.

By the Spectral Decomposition Theorem:

$$\mathbf{H} = \mathbf{P}\,\text{diag}(\lambda_1, \ldots, \lambda_n)\mathbf{P}^\top$$

where $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_n]$ and:

$$\mathbf{P}\mathbf{P}^\top = \mathbf{P}^\top\mathbf{P} = \mathbf{I}$$

For $j = 1, \ldots, n$, Expressing $\mathbf{y}$ in terms of the eigenvectors of $\mathbf{H}$, we have:

$$\mathbf{y} = \sum_{j=1}^{n} x_j \mathbf{p}_j \rightarrow x_j = \mathbf{p}_j^\top \mathbf{y}$$

By Spectral Decomposition, and then simplifying:

$$\mathbf{y}^\top\mathbf{H}\mathbf{y} = \mathbf{y}^\top\mathbf{P}\Lambda\mathbf{P}^\top\mathbf{y} = \mathbf{y}^\top \left(\sum_{j=1}^{n} \lambda_j \mathbf{p}_j \mathbf{p}_j^\top \right) \mathbf{y} = \sum_{j=1}^{n} \lambda_j \mathbf{y}^\top \mathbf{p}_j \mathbf{p}_j^\top \mathbf{y} = \sum_{j=1}^{n} \lambda_j x_j^2$$

Individually, we know x_j^2 is non-negative. And as given, we know λ_j is non-negative. Taken together, we know that each term $\lambda_j x_j^2$ is non-negative, such that their sum is non-negative as well. This means we have shown:

$$\mathbf{y}^\top\mathbf{H}\mathbf{y} \geq 0$$

Making the matrix $\mathbf{H}$ nonnegative definite.

Second direction complete!

Iff

Taken together, for H is a symmetric matrix, H is nonnegative definite $\iff$ all its eigenvalues are nonnegative.