R Demonstration – One-Way ANOVA

Objective: The purpose of this week's session is to demonstrate how to perform a one-way analysis of variance (ANOVA) in R. In the first part, we will use the built-in R functions to do this. In the second part, we will use the technique of partitioning the sum of squares to perform the ANOVA. In the third and final part, we will create a set of "dummy variables" and use them in a multiple regression to demonstrate the link between regression and ANOVA linear models.

Part I. Performing a one-way ANOVA using built-in R functions

NOTE: This part of the exercise assumes that you have downloaded the hypothetical dataset that records the flowering period of larkspur plants from 12 alpine meadow plots (see pages 292-293 of the Gotelli & Ellison text) and saved it in your `PCB6466` folder as a tab-delimited text file named *`ANOVA_data.txt`*. You also need to download the *`ANOVA.R`* script and save it in your `PCB6466` folder.

After starting R, change the directory to your `PCB6466` folder and open the *`ANOVA.R`* script. The first two lines of the script read and attach the larkspur dataset:

```
## read and attach the data
anova_data <- read.table("ANOVA_data.txt", header=T, row.names="ID")
attach(anova_data)
```

Next, we use the *`factor`* function to indicate that the variable named `GROUP` is a categorical variable:

```
## use the factor() function to tag GROUP as a categorical variable
factor(GROUP)
```

It is necessary for us to use the *`factor`* function in order to perform any ANOVA when the levels of the factor are originally coded as numbers. It defines the groups in our categorical variable properly. Generally, it is a good habit to use the *`factor`* function on all of your categorical variables in R.

Now we are ready to perform the one-way ANOVA using R's built-in functions. Since ANOVA is just another form of linear model, we will use the familiar *`lm`* function to create a linear model relating the response variable (`FLOWERING_PERIOD`) to the predictor variable (`GROUP`). Then we will use the *`anova`* function to produce summary statistics, including an ANOVA table, for our model. Here is the R code:

```
## use the lm() and anova() functions to execute the ANOVA
anova_model <- lm(FLOWERING_PERIOD ~ GROUP)
anova(anova_model)
```

The call to the `anova` function produces the following output:

```
Analysis of Variance Table

Response: FLOWERING_PERIOD
          Df  Sum Sq Mean Sq F value  Pr(>F)
GROUP      2 22.1667 11.0833  5.1154 0.03282 *
Residuals  9 19.5000  2.1667
---
```

This is a standard ANOVA table, and it is an almost identical match to the one presented in the Gotelli & Ellison text for this dataset (Table 10.3 on page 299).

Part II. Performing ANOVA by partitioning the sum of squares

Now that we have seen how easy it is to perform a one-way ANOVA using R's built-in functionality, we will use the technique of "partitioning the sum of squares" to dig a little more deeply into how the ANOVA process works. As discussed in the Gotelli & Ellison text, partitioning the sum of squares for a one-way ANOVA involves dividing the total variation into 2 components: the variation *among* groups and the variation *within* groups.

The **variation within groups** represents how much, on average, the observations within each group differ from their group mean. The sum of squares within groups (SS_{within}) is thus calculated by the following formula:

$$\boxed{SS_{within} = \sum_{i=1}^{a} \sum_{j=1}^{n} (Y_{ij} - \bar{Y}_i)^2}$$

To use this formula to calculate SS_{within} in R, we first assign the observations to their appropriate group using the familiar brackets notation to filter `FLOWERING_PERIOD` by group ("Unmanipulated", "Control", or "Treatment"):

```
## get the subset of the data for each group
Y_group1 <- FLOWERING_PERIOD[GROUP=="Unmanipulated"]
Y_group2 <- FLOWERING_PERIOD[GROUP=="Control"]
Y_group3 <- FLOWERING_PERIOD[GROUP=="Treatment"]
```

Next, we use the `levels` function (together with the `length` function) to determine the number of groups in our model, which we represent with the variable `a`. And because, for our simple one-way ANOVA, we assume that all the groups contain the same number of replicate observations, we can use the `length` function on any one of these groups to calculate the number of replicates, `n`:

```
## get the number of groups (a) and the number of replicates per group (n)
a <- length(levels(GROUP))
n <- length(Y_group1)
```

Now we need to calculate the means for each of the groups ($\bar{Y}_i$), as well as the "grand mean" for all of the observations ($\bar{Y}$):

```
## calculate the grand mean and the group means
grand_mean <- mean(FLOWERING_PERIOD)
group1_mean <- mean(Y_group1)
group2_mean <- mean(Y_group2)
group3_mean <- mean(Y_group3)
```

Finally, we can calculate the sum of squares within all of the groups and then sum these together to get the overall SS_{within}:

```
## calculate the sum of squares (SS) WITHIN the groups
SS_within1 = sum((Y_group1 - group1_mean)^2)
SS_within2 = sum((Y_group2 - group2_mean)^2)
SS_within3 = sum((Y_group3 - group3_mean)^2)
SS_within = SS_within1 + SS_within2 + SS_within3
```

The **variation among groups** represents how much each of the group means differs from the grand mean. When all of the groups contain the same number of replicate observations (n), as in the larkspur example, the sum of squares among groups (SS_{among}) can be calculated with the following formula:

$$\boxed{SS_{among} = n \cdot \sum_{i=1}^{a} (\bar{Y}_i - \bar{Y})^2}$$

In R, we use the following code to calculate the sum of squares among all of the groups and then sum these together to get the overall SS_{among}:

```
## calculate the sum of squares (SS) AMONG the groups
SS_among1 = n * sum((group1_mean - grand_mean)^2)
SS_among2 = n * sum((group2_mean - grand_mean)^2)
SS_among3 = n * sum((group3_mean - grand_mean)^2)
SS_among = SS_among1 + SS_among2 + SS_among3
```

To convert the SS_{within} and SS_{among} values calculated above to variance estimates (i.e. mean squares), we need to divide each sum of squares by its appropriate number of degrees of freedom (`df`). Since the degrees of freedom among groups equals *`(a-1)`* and the degrees of freedom within groups equals *`a(n-1)`*, we can use the following R code to calculate the mean square (MS) within and among groups:

```
## calculate the mean square (MS) values
df_among = (a-1)
df_within = a*(n-1)
MS_among = SS_among/df_among
MS_within = SS_within/df_within
```

Finally, we can use the values calculated for `MS_among` and `MS_within` to compute an F-ratio for our one-way ANOVA. Then we use the *`pf`* function to calculate a P-value for the F-ratio with `df_among` degrees of freedom in the numerator and `df_within` degrees of freedom in the denominator:

```
## calculate the F-ratio and corresponding P-value
F_ratio = MS_among/MS_within
P_value = pf(F_ratio, df_among, df_within, lower.tail=F)
```

The computed F-ratio of 5.115385 and the P-value of 0.03281756 are, again, nearly identical to those reported in Table 10.3 of the Gotelli & Ellison text.

Part III. Linking ANOVA to the regression model

As discussed in the lecture, it is possible to perform a one-way ANOVA using a multiple regression model containing "dummy variables". A **dummy variable** is a variable that can take on only one of two possible values: 0 or 1. To use this technique, we need to create a multiple regression model that contains *`(a-1)`* dummy variables (i.e. one less than the number of groups). Thus, for our larkspur dataset in which there are three groups, we would use the following multiple regression model:

$$\boxed{Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i}}$$

To create this multiple regression model in R, we first need to add 2 dummy variables (*`X1`* and *`X2`*) to our data frame:

```
## add "dummy variables" X1 and X2 to the data frame and initialize them to 0
anova_data$X1 <- numeric(length(GROUP))
anova_data$X2 <- numeric(length(GROUP))
```

As used above, the *`numeric`* function will initialize all of the values of both dummy variables to zero. This is fine for our "Unmanipulated" group, since we want both *`X1`* and *`X2`* to be zero for observations in this group. On the other hand, we want to indicate that an observation belongs to the "Control" group by setting *`X1`* equal to one and that an observation belongs to the "Treatment" group by setting *`X2`* equal to one. We thus use the following code to accomplish this:

```
## for the Unmanipulated group, X1=0 and X2=0; for the Control group, X1=1
anova_data$X1[GROUP=="Control"] <- 1

## for the Treatment group, X2=1
anova_data$X2[GROUP=="Treatment"] <- 1
```

After executing these lines of code, we can see that our dummy variables are set up properly by displaying the contents of the `anova_data` data frame:

```
> anova_data
            GROUP FLOWERING_PERIOD X1 X2
1  Unmanipulated               10  0  0
2  Unmanipulated               12  0  0
3  Unmanipulated               12  0  0
4  Unmanipulated               13  0  0
5        Control                9  1  0
6        Control               11  1  0
7        Control               11  1  0
8        Control               12  1  0
9      Treatment               12  0  1
10     Treatment               13  0  1
11     Treatment               15  0  1
12     Treatment               16  0  1
```

Now it is a simple matter to use the *`lm`* function to define a multiple regression model linking `FLOWERING_PERIOD` to these dummy variables, and then the *`summary`* function to summarize the results of the regression:

```
## now perform multiple regression with the dummy variables
regression_model <- lm(FLOWERING_PERIOD ~ anova_data$X1 + anova_data$X2)
summary(regression_model)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)     11.750      0.736  15.965 6.56e-08 ***
anova_data$X1   -1.000      1.041  -0.961   0.3618
anova_data$X2    2.250      1.041   2.162   0.0589 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.472 on 9 degrees of freedom
Multiple R-squared: 0.532,      Adjusted R-squared: 0.428
F-statistic: 5.115 on 2 and 9 DF,  p-value: 0.03282
```

Notice that, once again, the reported F-ratio and P-value for the multiple regression are identical to those listed in Table 10.3 of the Gotelli & Ellison text. To see how these values were calculated, recall that for a regression we can partition the total sum of squares into the SS from the regression and the SS from the residual errors. But to calculate these sums of squares, we first need to obtain the coefficients from our regression model and then use these to calculate the fitted values (i.e., the “Y_hats”):

```
## to perform an ANOVA for the regression, we first need the fitted values
coeffs <- regression_model$coefficients
Y_hats <- coeffs[1] + coeffs[2]*anova_data$X1 + coeffs[3]*anova_data$X2
```

Now we can calculate the sum of squares of the regression by squaring and summing the differences between the fitted values and the grand mean, and we can calculate the residual sum of squares by squaring and summing the differences between the observed values and the fitted values:

```
## calculate the SS for the regression (SSreg) and the residuals (SSresid)
SSreg <- sum((Y_hats - grand_mean)^2)
SSresid <- sum((FLOWERING_PERIOD - Y_hats)^2)
```

Finally, we again calculate our F-ratio as the ratio between our variances, only this time we use the mean square of the regression divided by the mean square residual:

```
## finally, calculate the F-ratio and P-value for the regression
F_ratio <- (SSreg/df_among)/(SSresid/df_within)
P_value <- pf(F_ratio, df_among, df_within, lower.tail=F)
```

As in the previous section of this exercise, the computed F-ratio of 5.115385 and the P-value of 0.03281756 are nearly identical to those reported in Table 10.3 of the Gotelli & Ellison text. Thus, it is clear that any one-way ANOVA analysis can be represented as a multiple regression model.

Part IV: Analysis using OpenBUGS (means format)

This portion of the code creates a Bayesian ANOVA similar in format to the approaches discussed in parts I and II.

```
x <- c(1,1,1,1,2,2,2,2,3,3,3,3)
n_obs <- length(x)
z <- FLOWERING_PERIOD
## Call package
library(R2OpenBUGS)
### Fitting the model
# Write model
Anovam<-function()
## Priors
{
 for (i in 1:3)
   {
     a[i] ~ dnorm(0.0,1.0E-6)
   }
 tau ~ dgamma(0.001,0.001)
## Likelihood
 for (i in 1:n)
   {
   mean[i] <- a[x[i]]
   Y[i]  ~ dnorm(mean[i],tau)
   }
  d21 <- a[2]-a[1]
  d31 <- a[3]-a[1]
  d32 <- a[3]-a[2]
  }
write.model(Anovam, "Anovam.txt")

# Bundle data
win.data <- list(Y=z,x=x, n=n_obs)
# Inits function
```

```
inits <- function()
   list(a=runif(3,1,10),tau=runif(1))
# Parameters to estimate
params <- c("a","d21","d31","d32")
# MCMC settings
nc = 3
ni=10000
nb=1000
nt=10
# Start Gibbs sampler
out <- bugs(data = win.data, inits = inits, parameters = params, model = 
"Anovam.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni, digits=5, codaPkg=T)
library(coda)
reg.coda<-read.bugs(out)
results<-summary(reg.coda)
results
```

As usual we expect the results of a Bayesian analysis with uninformed priors to give numerical estimates that are commensurate with the frequentist's, but also information to quantify the exact probability of numerical hypothesis associated with our data.

Part V: Analysis using OpenBUGS (regression format)

This section uses OpenBUGS to calculate the ANOVA using a Bayesian regression format and a reference class.

```
### Fitting the model
# Write model
Anovar<-function()
## Priors
{
 reference ~ dnorm(0,1.0E-6)
 for (i in 2:3)
   {
     d[i] ~ dnorm(0.0,1.0E-6)
   }
 d[1] <- 0
 tau ~ dgamma(0.001,0.001)
## Likelihood
 for (i in 1:n)
   {
   mean[i] <- reference + d[x[i]]
   Y[i]  ~ dnorm(mean[i],tau)
   }
  }
write.model(Anovar, "Anovar.txt")
# Bundle data
win.data <- list(Y=z,x=x, n=n_obs)
# Inits function
inits <- function()
   list(reference= runif(1,0,30), d= c(NA,0,0), tau=runif(1))
# Parameters to estimate
params <- c("reference","d")
# MCMC settings
nc = 3
ni=10000
nb=1000
nt=10

# Start Gibbs sampler
```

```
out <- bugs(data = win.data, inits = inits, parameters = params, model =
"Anovar.txt",
n.thin = nt, n.chains = nc, n.burnin = nb, n.iter = ni,  digits=5, codaPkg=T)
reg.coda<-read.bugs(out)
results<-summary(reg.coda)
results
```

To interpret the results, remember that you are using one of the treatments as a reference to compare the others too it. As usual we end by detaching the data.

```
## detach the data
detach(anova_data)
```