Lab 9 Intro to ANOVA

[brouwern@gmail.com](mailto:brouwern@gmail.com)

November 3, 2016

## Create data

mass<-c(472.5114,722.7774,548.6796,464.3168,577.2037,406.7964,555.8052,657.5226,734.7416,919.2983,665.6099,570.3627,429.5572,520.9726,346.0116,639.3096,544.8374,657.6512,802.2045,443.339,545.9901,791.3983,884.5803,763.2926,782.1005,840.2083,417.7051,735.8959,759.7498,877.0629)  
n <- length(mass)/3  
diet <- c(rep("Hi.Hi",n),rep("Hi.Lo",n),rep("Lo.Hi",n))  
  
antlers <- data.frame(antler.mass = mass,  
 diet = diet)  
  
rm(list = c("mass", "diet","n"))

# plot3means

# plotTukeysHSD

## Part 1: Summary stats & plotting means

## 1) Summary stats on all data

# total sample size  
dim(antlers)

## [1] 30 2

n.total <- length(antlers$antler.mass)  
  
#mean  
summary(antlers)

## antler.mass diet   
## Min. :346.0 Hi.Hi:10   
## 1st Qu.:526.9 Hi.Lo:10   
## Median :648.4 Lo.Hi:10   
## Mean :635.9   
## 3rd Qu.:762.4   
## Max. :919.3

mean(antlers$antler.mass)

## [1] 635.9164

#variance  
var(antlers$antler.mass)

## [1] 25767.67

#stdev  
mass.sd <- sd(antlers$antler.mass)

## 2) SE for \*\* all \*\* data

#Using save sd  
mass.se <- mass.sd/sqrt(n.total)  
  
#Using raw data  
mass.se <- sd(antlers$antler.mass)/sqrt(length(antlers$antler.mass))

## 3) 95% CI for all data

1.96\*mass.se

## [1] 57.44246

## 4) Calcualte summary stats for each group

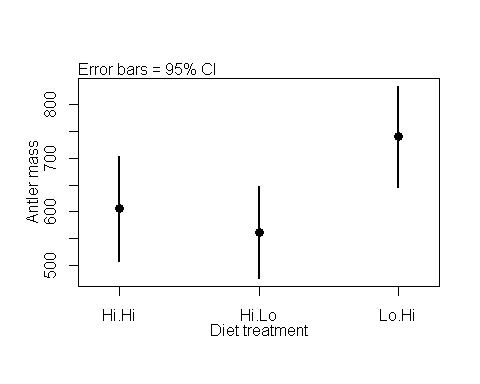
# Means  
mass.3.means <- tapply(antlers$antler.mass,  
 antlers$diet,  
 mean)  
  
#SD  
mass.3.sd <- tapply(antlers$antler.mass,  
 antlers$diet,  
 sd)  
  
#sample size n  
mass.3.n <- tapply(antlers$antler.mass,  
 antlers$diet,  
 length)

## 5) Calcualte the 3 SE values

mass.3.se <- mass.3.sd/sqrt(mass.3.n)

## 6) Plot the 3 mean valeus

#Using the levels command  
antler.groups <- levels(antlers$diet)  
  
#By hand  
antler.groups <- c("Hi.Hi", "Hi.Lo", "Lo.Hi")  
  
plot3means(means = mass.3.means,  
 se = mass.3.se,  
 categories = antler.groups,x.axis.label = "Diet treatment",  
 y.axis.label = "Antler mass"  
 )



## [1] "This function plots approximate 95% confidence intervals based on standard error imput by the user. DO not enter 95% CIs"

## Part 2: Omnibus ANOVA test

## 7 & 8) Build the null and alternative model

model.null <- lm(antler.mass ~ 1,   
 data = antlers)  
model.alt <- lm(antler.mass ~ diet,   
 data = antlers)

## 9-12) Conduct the Omnibus test and get the ANOVA output

Produce the ANOVA table

anova(model.null,model.alt)

## Analysis of Variance Table  
##   
## Model 1: antler.mass ~ 1  
## Model 2: antler.mass ~ diet  
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 29 747262   
## 2 27 575719 2 171543 4.0225 0.02958 \*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

## OPTIONAL:

If you are curious and things are going well, do the optional activities

### The summary command on an lm() object

summary(model.null)

##   
## Call:  
## lm(formula = antler.mass ~ 1, data = antlers)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -289.9 -109.0 12.5 126.5 283.4   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 635.92 29.31 21.7 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 160.5 on 29 degrees of freedom

summary(model.alt)

##   
## Call:  
## lm(formula = antler.mass ~ diet, data = antlers)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -322.09 -103.31 14.16 99.22 313.33   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 605.97 46.18 13.123 3.12e-13 \*\*\*  
## dietHi.Lo -43.98 65.30 -0.673 0.5064   
## dietLo.Hi 133.83 65.30 2.049 0.0503 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 146 on 27 degrees of freedom  
## Multiple R-squared: 0.2296, Adjusted R-squared: 0.1725   
## F-statistic: 4.022 on 2 and 27 DF, p-value: 0.02958

### The anova command on a single lm() object

anova(model.alt)

## Analysis of Variance Table  
##   
## Response: antler.mass  
## Df Sum Sq Mean Sq F value Pr(>F)   
## diet 2 171543 85772 4.0225 0.02958 \*  
## Residuals 27 575719 21323   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

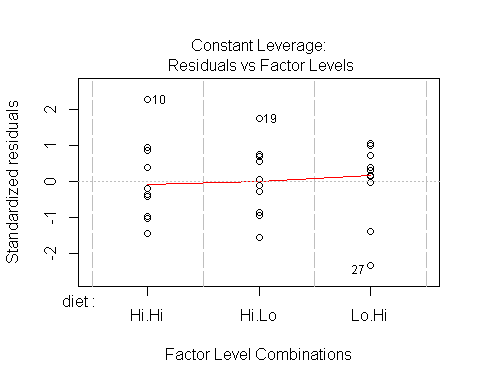
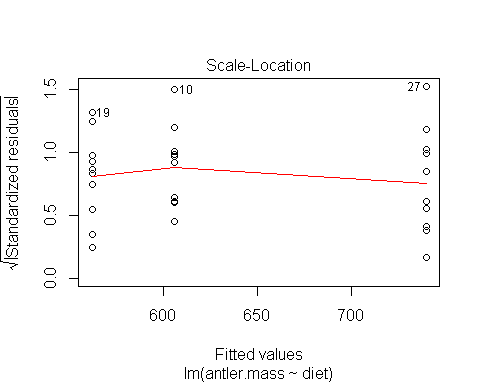
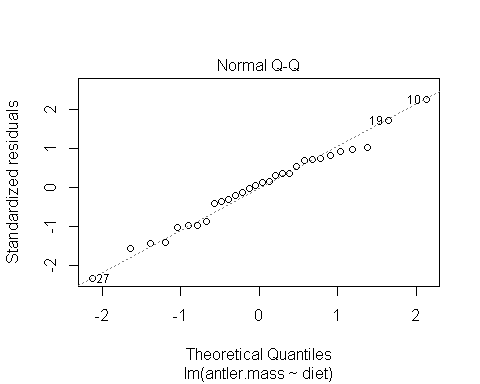
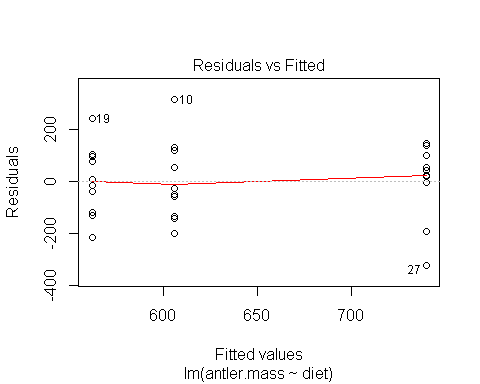
### Change how model is defined

model.alt.2 <- lm(antler.mass ~-1 + diet, data = antlers)  
  
summary(model.alt.2)

##   
## Call:  
## lm(formula = antler.mass ~ -1 + diet, data = antlers)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -322.09 -103.31 14.16 99.22 313.33   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## dietHi.Hi 605.97 46.18 13.12 3.12e-13 \*\*\*  
## dietHi.Lo 561.99 46.18 12.17 1.80e-12 \*\*\*  
## dietLo.Hi 739.80 46.18 16.02 2.59e-15 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 146 on 27 degrees of freedom  
## Multiple R-squared: 0.9553, Adjusted R-squared: 0.9503   
## F-statistic: 192.3 on 3 and 27 DF, p-value: < 2.2e-16

### Plotting model diagnostics

plot(model.alt)



## Part 3: Pariwise comparisons

## 13-15) Get pairwise p-valeus

pairwise.t.test(x = antlers$antler.mass,  
 g = antlers$diet,  
 p.adjust.method = "none")

##   
## Pairwise comparisons using t tests with pooled SD   
##   
## data: antlers$antler.mass and antlers$diet   
##   
## Hi.Hi Hi.Lo  
## Hi.Lo 0.506 -   
## Lo.Hi 0.050 0.011  
##   
## P value adjustment method: none

## Part 4: Pairwise comparisons, correcting for multiple comparisons, & effect sizes

## 17) Build model with aov() function

model.alt.aov <- aov(antler.mass ~ diet, data = antlers)

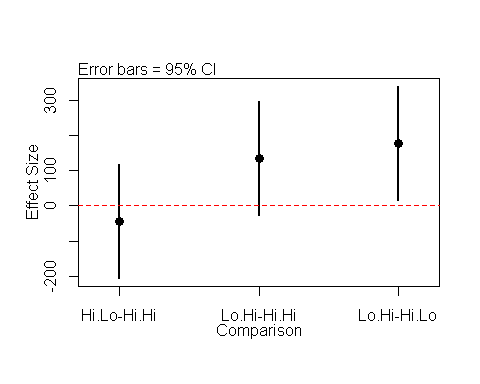
## 18) Get p-values with Tukey's HSD

TukeyHSD(model.alt.aov)

## Tukey multiple comparisons of means  
## 95% family-wise confidence level  
##   
## Fit: aov(formula = antler.mass ~ diet, data = antlers)  
##   
## $diet  
## diff lwr upr p adj  
## Hi.Lo-Hi.Hi -43.97973 -205.89517 117.9357 0.7807326  
## Lo.Hi-Hi.Hi 133.83308 -28.08236 295.7485 0.1198088  
## Lo.Hi-Hi.Lo 177.81281 15.89737 339.7282 0.0292046

## 19) Plot effect sizes

tukey.out <- TukeyHSD(model.alt.aov)  
  
plotTukeysHSD(tukey.out)  
abline(h = 0, col = 2, lty = 2)



## Make new data

# Make circumference data

circum <- c(94.85394,83.21354,64.94166,96.9141,77.03692,113.2507,90.95746,95.83833,82.66041,97.46359,91.61469,96.4177,100.5873,99.3556,94.23969,57.94891,134.9324,95.01268,128.2977,129.307,70.39667,133.7504,121.0767,105.3898,86.81835,106.1906,136.0613,74.95854,112.6994,110.5714)  
  
  
n <- length(circum)/3  
diet <- c(rep("Hi.Hi",n),rep("Hi.Lo",n),rep("Lo.Hi",n))  
  
df.circum <- data.frame(antler.circum = circum,  
 diet = diet)  
  
rm(list = c("mass", "diet","n"))

## Warning in rm(list = c("mass", "diet", "n")): object 'mass' not found

df.circum

## antler.circum diet  
## 1 94.85394 Hi.Hi  
## 2 83.21354 Hi.Hi  
## 3 64.94166 Hi.Hi  
## 4 96.91410 Hi.Hi  
## 5 77.03692 Hi.Hi  
## 6 113.25070 Hi.Hi  
## 7 90.95746 Hi.Hi  
## 8 95.83833 Hi.Hi  
## 9 82.66041 Hi.Hi  
## 10 97.46359 Hi.Hi  
## 11 91.61469 Hi.Lo  
## 12 96.41770 Hi.Lo  
## 13 100.58730 Hi.Lo  
## 14 99.35560 Hi.Lo  
## 15 94.23969 Hi.Lo  
## 16 57.94891 Hi.Lo  
## 17 134.93240 Hi.Lo  
## 18 95.01268 Hi.Lo  
## 19 128.29770 Hi.Lo  
## 20 129.30700 Hi.Lo  
## 21 70.39667 Lo.Hi  
## 22 133.75040 Lo.Hi  
## 23 121.07670 Lo.Hi  
## 24 105.38980 Lo.Hi  
## 25 86.81835 Lo.Hi  
## 26 106.19060 Lo.Hi  
## 27 136.06130 Lo.Hi  
## 28 74.95854 Lo.Hi  
## 29 112.69940 Lo.Hi  
## 30 110.57140 Lo.Hi

# Make circumference data

beam <- c(335.6464,410.2204,425.6838,355.1535,346.7697,399.8559,425.4827,447.6792,397.116,428.6749,406.034,380.342,437.5513,444.8916,437.9827,391.446,425.6301,404.2163,461.0517,568.163,457.6777,459.0168,270.5814,440.9724,412.0543,428.0725,247.2619,386.4118,423.5378,505.5221)  
  
  
n <- length(beam)/3  
diet <- c(rep("Hi.Hi",n),rep("Hi.Lo",n),rep("Lo.Hi",n))  
  
df.beam <- data.frame(antler.beam = beam,  
 diet = diet)  
  
rm(list = c("beam", "diet","n"))  
df.beam

## antler.beam diet  
## 1 335.6464 Hi.Hi  
## 2 410.2204 Hi.Hi  
## 3 425.6838 Hi.Hi  
## 4 355.1535 Hi.Hi  
## 5 346.7697 Hi.Hi  
## 6 399.8559 Hi.Hi  
## 7 425.4827 Hi.Hi  
## 8 447.6792 Hi.Hi  
## 9 397.1160 Hi.Hi  
## 10 428.6749 Hi.Hi  
## 11 406.0340 Hi.Lo  
## 12 380.3420 Hi.Lo  
## 13 437.5513 Hi.Lo  
## 14 444.8916 Hi.Lo  
## 15 437.9827 Hi.Lo  
## 16 391.4460 Hi.Lo  
## 17 425.6301 Hi.Lo  
## 18 404.2163 Hi.Lo  
## 19 461.0517 Hi.Lo  
## 20 568.1630 Hi.Lo  
## 21 457.6777 Lo.Hi  
## 22 459.0168 Lo.Hi  
## 23 270.5814 Lo.Hi  
## 24 440.9724 Lo.Hi  
## 25 412.0543 Lo.Hi  
## 26 428.0725 Lo.Hi  
## 27 247.2619 Lo.Hi  
## 28 386.4118 Lo.Hi  
## 29 423.5378 Lo.Hi  
## 30 505.5221 Lo.Hi

### Spread data

spread <- c(271.4356,291.3746,346.3026,345.6056,234.1441,339.7039,291.5156,275.9856,223.7703,316.9263,375.1109,384.1469,396.6894,272.4158,356.8185,393.6393,412.3185,425.7551,390.7663,381.4302,341.7604,350.6951,306.0166,354.8825,312.0478,459.7191,415.1186,321.8994,281.539,372.0439)  
  
  
n <- length(spread)/3  
diet <- c(rep("Hi.Hi",n),rep("Hi.Lo",n),rep("Lo.Hi",n))  
  
df.spread <- data.frame(antler.spread = spread,  
 diet = diet)  
  
rm(list = c("spread", "diet","n"))  
df.spread

## antler.spread diet  
## 1 271.4356 Hi.Hi  
## 2 291.3746 Hi.Hi  
## 3 346.3026 Hi.Hi  
## 4 345.6056 Hi.Hi  
## 5 234.1441 Hi.Hi  
## 6 339.7039 Hi.Hi  
## 7 291.5156 Hi.Hi  
## 8 275.9856 Hi.Hi  
## 9 223.7703 Hi.Hi  
## 10 316.9263 Hi.Hi  
## 11 375.1109 Hi.Lo  
## 12 384.1469 Hi.Lo  
## 13 396.6894 Hi.Lo  
## 14 272.4158 Hi.Lo  
## 15 356.8185 Hi.Lo  
## 16 393.6393 Hi.Lo  
## 17 412.3185 Hi.Lo  
## 18 425.7551 Hi.Lo  
## 19 390.7663 Hi.Lo  
## 20 381.4302 Hi.Lo  
## 21 341.7604 Lo.Hi  
## 22 350.6951 Lo.Hi  
## 23 306.0166 Lo.Hi  
## 24 354.8825 Lo.Hi  
## 25 312.0478 Lo.Hi  
## 26 459.7191 Lo.Hi  
## 27 415.1186 Lo.Hi  
## 28 321.8994 Lo.Hi  
## 29 281.5390 Lo.Hi  
## 30 372.0439 Lo.Hi

## x) Subset data into eachseperate group

df.Hi.Hi <- subset(antlers,  
 diet == "Hi.Hi")  
df.Hi.Lo <- subset(antlers,  
 diet == "Hi.Lo")  
df.Lo.Lo <- subset(antlers,  
 diet == "Lo.Lo")

## x) Carry out a t-test on each combination

t.test.HHvsHL <- t.test(df.Hi.Hi$antler.mass,  
 df.Hi.Lo$antler.mass)  
  
  
t.test.HHvsHL <- t.test(df.Hi.Hi$antler.mass,  
 df.Hi.Lo$antler.mass)