Lab 9: All Data

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

November 6, 2016

# means = 3 mean values calcualted from raw data  
# se = 3 standard errors  
# categories = 3 categorical / factors that group the data  
# x.axis.label = what should be plotted on the x axis  
# y.axis.label = what should be plotted on the y axis  
  
  
#### The function STARTS here ####  
plotTukeysHSD <- function(tukey.out,  
 x.axis.label = "Comparison",  
 y.axis.label = "Effect Size",  
 axis.adjust = 0){  
   
 tukey.out <- as.data.frame(tukey.out[[1]])  
 means <- tukey.out$diff  
 categories <- row.names(tukey.out)  
 groups <- length(categories)  
 ci.low <- tukey.out$lwr  
 ci.up <- tukey.out$upr   
   
 # reset plot window  
 #par(mfrow = c(1,1),   
 # mar = c(3.5,3,2,1))  
   
 # calculate values for plotting limits   
 y.max <- max(ci.up) +   
 max(ci.up)\*axis.adjust  
 y.min <- min(ci.low) -   
 max(ci.low)\*axis.adjust  
   
 if(groups == 2){ x.values <- c(0.25, 0.5)}  
 if(groups == 3){ x.values <- c(0.25, 0.5,0.75)}  
   
 x.axis.min <- min(x.values)-0.05  
 x.axis.max <- max(x.values)+0.05  
   
 x.limits <- c(x.axis.min,x.axis.max)  
   
 #Plot means  
 plot(means ~ x.values,  
 xlim = x.limits,  
 ylim = c(y.min,y.max),  
 xaxt = "n",  
 xlab = "",  
 ylab = "",  
 cex = 1.25,  
 pch = 16)  
   
 axis(side = 1,   
 at = x.values,  
 labels = categories,  
 )  
   
 #Plot upper error bar   
 lwd. <- 2  
 arrows(y0 = means,  
 x0 = x.values,  
 y1 = ci.up,  
 x1 = x.values,  
 length = 0,  
 lwd = lwd.)  
   
 #Plot lower error bar  
 arrows(y0 = means,  
 x0 = x.values,  
 y1 = ci.low,  
 x1 = x.values,  
 length = 0,  
 lwd = lwd.)   
   
 mtext(text = x.axis.label,side = 1,line = 1.75)  
 mtext(text = y.axis.label,side = 2,line = 1.95)  
 mtext(text = "Error bars = 95% CI",side = 3,line = 0,adj = 0)  
   
   
}

## Outline

* Make 4 datasets of different antler measurements (mass, basal circumference, main beam length, inside spread)
* ANOVA omnibus F-test
* Pairwise t-tests
* TukeyHSD tests
* Plot effect sizes from TukeyHSD

# Preliminaries: Make datasets

### Datset 1) Antler mass 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))  
df.mass <- data.frame(antler.mass = mass,  
 diet = diet)  
rm(list = c("mass", "diet","n"))

### Datset 2) Antler mass 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("circum", "diet","n"))

### Datset 3) Antler beam length 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"))

### Datset 4) Antler 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"))

# Data analysis: Mass data

### Summary stats for all mass data

#### Basic summary stats

mean(df.mass$antler.mass)

## [1] 635.9164

var(mean(df.mass$antler.mass))

## [1] NA

sd.mass <- sd(df.mass$antler.mass)  
n.mass <- length(mean(df.mass$antler.mass))

#### Standard errors

sd.mass/sqrt(n.mass)

## [1] 160.5231

### Summary stats by diet treatment

Use tapply to get means and standard deviations by group.

#### Calcualte means with tapply

mean.mass <- tapply(df.mass$antler.mass,  
 df.mass$diet,  
 FUN = mean)

#### Calcualte sd with tapply

sd.mass <- tapply(df.mass$antler.mass,  
 df.mass$diet,  
 FUN = sd)

#### Calcualte sample size with tapply

n.mass <- tapply(df.mass$antler.mass,  
 df.mass$diet,  
 FUN = length)

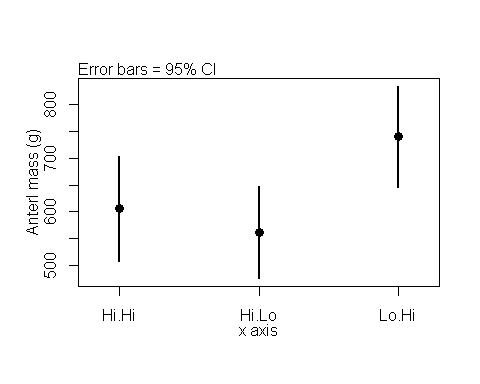
#### Calcualte SD

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

### Plot the means

This requires the plot3means function from <https://rpubs.com/brouwern/plot3means>

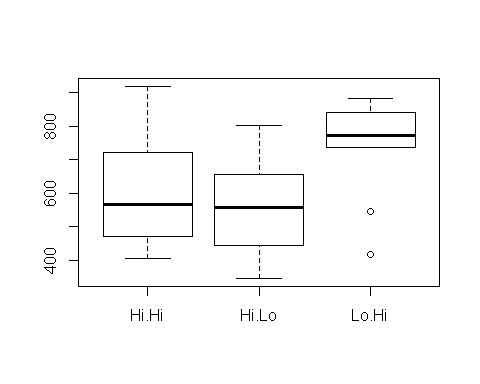
categories <- levels(df.mass$diet)  
  
plot3means(means = mean.mass,  
 se = se.mass,  
 categories = categories,  
 y.axis.label = "Anterl mass (g)"  
 )



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

### Plot raw data

boxplot(antler.mass ~ diet,df.mass)



### ANOVA omnibus F-test

Build models

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

Compare models to get F statistic and p-values

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

Save model output

mass.anova.table <- anova(model.null, model.alt)

### Pairwise t-test

summary(df.mass)

Make table of pairwise t-test

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

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

Save output

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

### TukeyHSD p-values

Fit model with aov() instead of lm()

mass.aov <- aov(antler.mass ~ diet, data = df.mass)

Calcaulte corrected p valeus and effect sizes

TukeyHSD(mass.aov)

## Tukey multiple comparisons of means  
## 95% family-wise confidence level  
##   
## Fit: aov(formula = antler.mass ~ diet, data = df.mass)  
##   
## $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

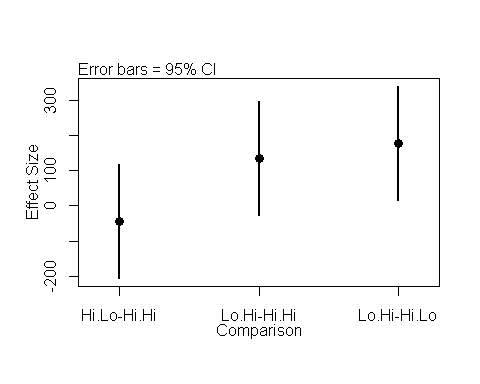
### Plot effect sizes from TukeyHSD

Save output of TukeyHSD to an object

mass.tuk <- TukeyHSD(mass.aov)

Use plotTukeyHSD from <https://rpubs.com/brouwern/plotTukeyHSD>

plotTukeysHSD(mass.tuk)



# Circumference data

` # Data analysis

### Summary stats for all data

#### Basic summary stats

mean(df.circum$antler.circum)

## [1] 99.42525

var(mean(df.circum$antler.circum))

## [1] NA

sd.circum <- sd(df.circum$antler.circum)  
n.circum <- length(mean(df.circum$antler.circum))

#### Standard errors

sd.circum/sqrt(n.circum)

## [1] 20.62602

### Summary stats by diet treatment

Use tapply to get means and standard deviations by group.

#### Calcualte means with tapply

mean.circum <- tapply(df.circum$antler.circum,  
 df.circum$diet,  
 FUN = mean)

#### Calcualte sd with tapply

sd.circum <- tapply(df.circum$antler.circum,  
 df.circum$diet,  
 FUN = sd)

#### Calcualte sample size with tapply

n.circum <- tapply(df.circum$antler.circum,  
 df.circum$diet,  
 FUN = length)

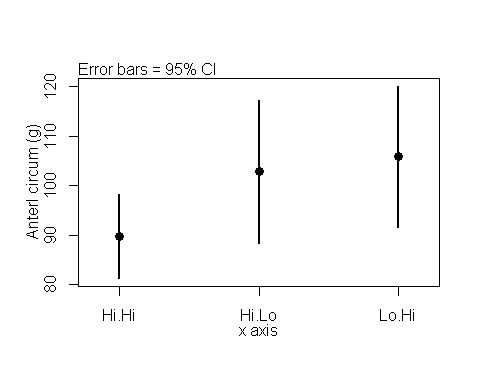
#### Calcualte SD

se.circum <- sd.circum/sqrt(n.circum)

### Plot the means

This requires the plot3means function from <https://rpubs.com/brouwern/plot3means>

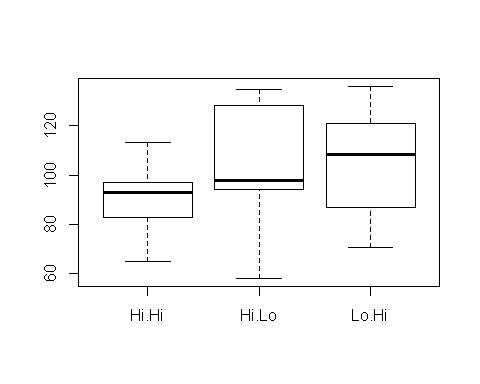
categories <- levels(df.circum$diet)  
  
plot3means(means = mean.circum,  
 se = se.circum,  
 categories = categories,  
 y.axis.label = "Anterl circum (g)"  
 )



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

### Plot raw data

boxplot(antler.circum ~ diet,df.circum)



### ANOVA omnibus F-test

Build models

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

Compare models to get F statistic and p-values

anova(model.null, model.alt)

## Analysis of Variance Table  
##   
## Model 1: antler.circum ~ 1  
## Model 2: antler.circum ~ diet  
## Res.Df RSS Df Sum of Sq F Pr(>F)  
## 1 29 12338   
## 2 27 10877 2 1460.5 1.8127 0.1825

Save model output

circum.anova.table <- anova(model.null, model.alt)

### Pairwise t-test

summary(df.circum)

Make table of pairwise t-test

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

##   
## Pairwise comparisons using t tests with pooled SD   
##   
## data: df.circum$antler.circum and df.circum$diet   
##   
## Hi.Hi Hi.Lo  
## Hi.Lo 0.157 -   
## Lo.Hi 0.084 0.739  
##   
## P value adjustment method: none

Save output

circum.pairwise.p <- pairwise.t.test(x = df.circum$antler.circum,  
 g = df.circum$diet,  
 p.adjust.method = "none")

### TukeyHSD p-values

Fit model with aov() instead of lm()

circum.aov <- aov(antler.circum ~ diet, data = df.circum)

Calcaulte corrected p valeus and effect sizes

TukeyHSD(circum.aov)

## Tukey multiple comparisons of means  
## 95% family-wise confidence level  
##   
## Fit: aov(formula = antler.circum ~ diet, data = df.circum)  
##   
## $diet  
## diff lwr upr p adj  
## Hi.Lo-Hi.Hi 13.058302 -9.197247 35.31385 0.3280747  
## Lo.Hi-Hi.Hi 16.078251 -6.177298 38.33380 0.1915525  
## Lo.Hi-Hi.Lo 3.019949 -19.235600 25.27550 0.9396505

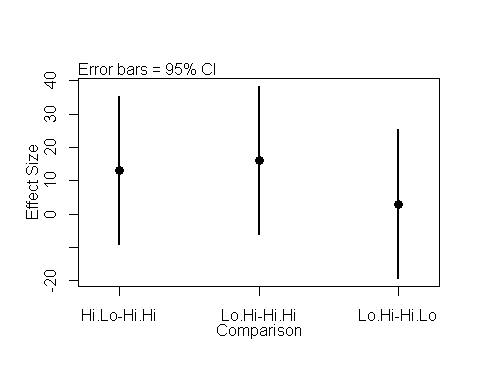
### Plot effect sizes from TukeyHSD

Save output of TukeyHSD to an object

circum.tuk <- TukeyHSD(circum.aov)

Use plotTukeyHSD from <https://rpubs.com/brouwern/plotTukeyHSD>

plotTukeysHSD(circum.tuk)



# Antler main beam data

# Data analysis

### Summary stats for all data

#### Basic summary stats

mean(df.beam$antler.beam)

## [1] 412.0233

var(mean(df.beam$antler.beam))

## [1] NA

sd.beam <- sd(df.beam$antler.beam)  
n.beam <- length(mean(df.beam$antler.beam))

#### Standard errors

sd.beam/sqrt(n.beam)

## [1] 61.19076

### Summary stats by diet treatment

Use tapply to get means and standard deviations by group.

#### Calcualte means with tapply

mean.beam <- tapply(df.beam$antler.beam,  
 df.beam$diet,  
 FUN = mean)

#### Calcualte sd with tapply

sd.beam <- tapply(df.beam$antler.beam,  
 df.beam$diet,  
 FUN = sd)

#### Calcualte sample size with tapply

n.beam <- tapply(df.beam$antler.beam,  
 df.beam$diet,  
 FUN = length)

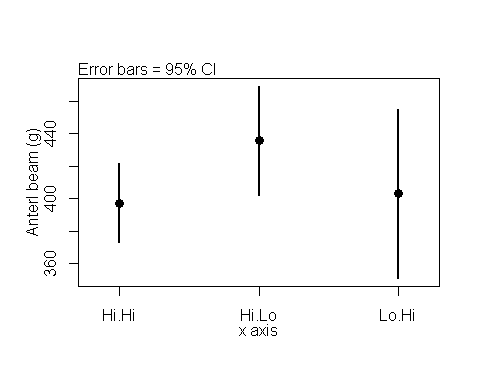
#### Calcualte SD

se.beam <- sd.beam/sqrt(n.beam)

### Plot the means

This requires the plot3means function from <https://rpubs.com/brouwern/plot3means>

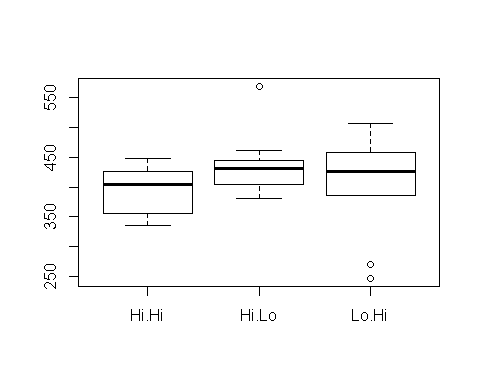
categories <- levels(df.beam$diet)  
  
plot3means(means = mean.beam,  
 se = se.beam,  
 categories = categories,  
 y.axis.label = "Anterl beam (g)"  
 )



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

### Plot raw data

boxplot(antler.beam ~ diet,df.beam)



### ANOVA omnibus F-test

Build models

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

Compare models to get F statistic and p-values

anova(model.null, model.alt)

## Analysis of Variance Table  
##   
## Model 1: antler.beam ~ 1  
## Model 2: antler.beam ~ diet  
## Res.Df RSS Df Sum of Sq F Pr(>F)  
## 1 29 108585   
## 2 27 99981 2 8603.7 1.1617 0.3281

Save model output

beam.anova.table <- anova(model.null, model.alt)

### Pairwise t-test

summary(df.beam)

Make table of pairwise t-test

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

##   
## Pairwise comparisons using t tests with pooled SD   
##   
## data: df.beam$antler.beam and df.beam$diet   
##   
## Hi.Hi Hi.Lo  
## Hi.Lo 0.17 -   
## Lo.Hi 0.83 0.24   
##   
## P value adjustment method: none

Save output

beam.pairwise.p <- pairwise.t.test(x = df.beam$antler.beam,  
 g = df.beam$diet,  
 p.adjust.method = "none")

### TukeyHSD p-values

Fit model with aov() instead of lm()

beam.aov <- aov(antler.beam ~ diet, data = df.beam)

Calcaulte corrected p valeus and effect sizes

TukeyHSD(beam.aov)

## Tukey multiple comparisons of means  
## 95% family-wise confidence level  
##   
## Fit: aov(formula = antler.beam ~ diet, data = df.beam)  
##   
## $diet  
## diff lwr upr p adj  
## Hi.Lo-Hi.Hi 38.50262 -28.97224 105.97748 0.3476276  
## Lo.Hi-Hi.Hi 5.88262 -61.59224 73.35748 0.9745943  
## Lo.Hi-Hi.Lo -32.62000 -100.09486 34.85486 0.4641737

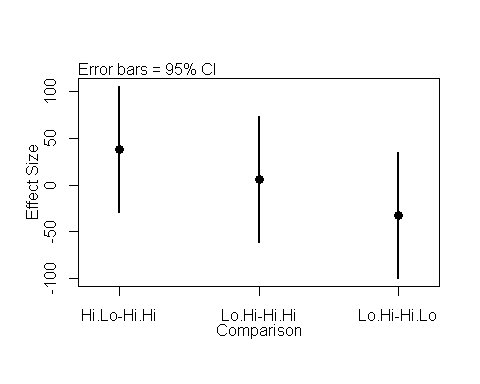
### Plot effect sizes from TukeyHSD

Save output of TukeyHSD to an object

beam.tuk <- TukeyHSD(beam.aov)

Use plotTukeyHSD from <https://rpubs.com/brouwern/plotTukeyHSD>

plotTukeysHSD(beam.tuk)



# Antler Spread Data

# Data analysis

### Summary stats for all data

#### Basic summary stats

mean(df.spread$antler.spread)

## [1] 341.3859

var(mean(df.spread$antler.spread))

## [1] NA

sd.spread <- sd(df.spread$antler.spread)  
n.spread <- length(mean(df.spread$antler.spread))

#### Standard errors

sd.spread/sqrt(n.spread)

## [1] 57.77433

### Summary stats by diet treatment

Use tapply to get means and standard deviations by group.

#### Calcualte means with tapply

mean.spread <- tapply(df.spread$antler.spread,  
 df.spread$diet,  
 FUN = mean)

#### Calcualte sd with tapply

sd.spread <- tapply(df.spread$antler.spread,  
 df.spread$diet,  
 FUN = sd)

#### Calcualte sample size with tapply

n.spread <- tapply(df.spread$antler.spread,  
 df.spread$diet,  
 FUN = length)

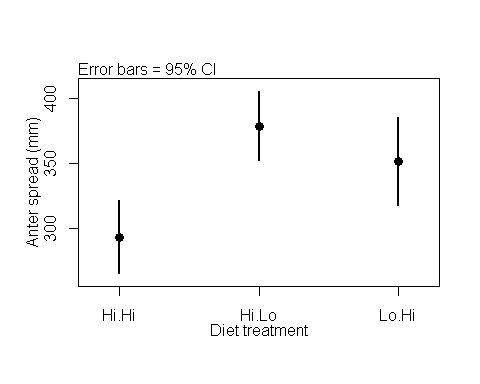
#### Calcualte SD

se.spread <- sd.spread/sqrt(n.spread)

### Plot the means

This requires the plot3means function from <https://rpubs.com/brouwern/plot3means>

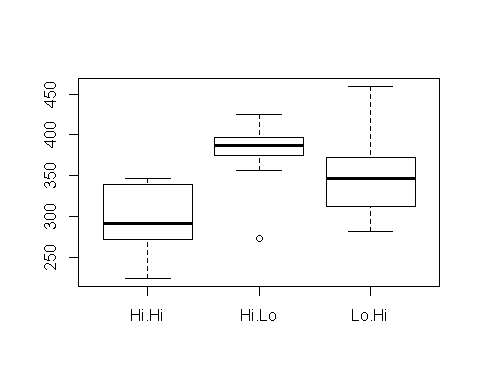
par(mfrow = c(1,1))  
categories <- levels(df.spread$diet)  
plot3means(means = mean.spread,  
 se = se.spread,  
 categories = categories,  
 x.axis.label = "Diet treatment",  
 y.axis.label = "Anter spread (mm)",axis.adjust = 0.01  
 )



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

### Plot raw data

boxplot(antler.spread ~ diet,df.spread)



### ANOVA omnibus F-test

Build models

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

Compare models to get F statistic and p-values

anova(model.null, model.alt)

## Analysis of Variance Table  
##   
## Model 1: antler.spread ~ 1  
## Model 2: antler.spread ~ diet  
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 29 96798   
## 2 27 58919 2 37879 8.6793 0.001228 \*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Save model output

spread.anova.table <- anova(model.null, model.alt)

### Pairwise t-test

Make table of pairwise t-test

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

##   
## Pairwise comparisons using t tests with pooled SD   
##   
## data: df.spread$antler.spread and df.spread$diet   
##   
## Hi.Hi Hi.Lo   
## Hi.Lo 0.00036 -   
## Lo.Hi 0.00998 0.20172  
##   
## P value adjustment method: none

Save output

spread.pairwise.p <- pairwise.t.test(x = df.spread$antler.spread,  
 g = df.spread$diet,  
 p.adjust.method = "none")

### TukeyHSD p-values

Fit model with aov() instead of lm()

spread.aov <- aov(antler.spread ~ diet, data = df.spread)

Calcaulte corrected p valeus and effect sizes

TukeyHSD(spread.aov)

## Tukey multiple comparisons of means  
## 95% family-wise confidence level  
##   
## Fit: aov(formula = antler.spread ~ diet, data = df.spread)  
##   
## $diet  
## diff lwr upr p adj  
## Hi.Lo-Hi.Hi 85.23267 33.435024 137.0303 0.0010122  
## Lo.Hi-Hi.Hi 57.89582 6.098174 109.6935 0.0261561  
## Lo.Hi-Hi.Lo -27.33685 -79.134496 24.4608 0.4027839

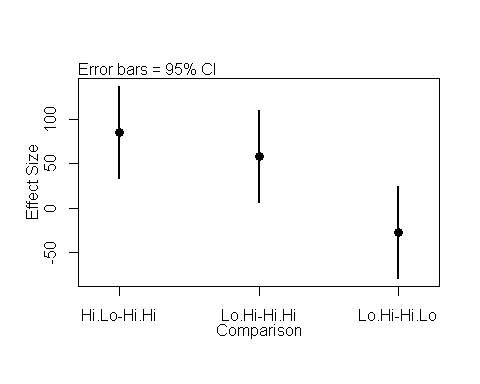
### Plot effect sizes from TukeyHSD

Save output of TukeyHSD to an object

spread.tuk <- TukeyHSD(spread.aov)

Use plotTukeyHSD from <https://rpubs.com/brouwern/plotTukeyHSD>

plotTukeysHSD(spread.tuk)



All output

mass.anova.table

## 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

mass.pairwise.p

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

mass.aov

## Call:  
## aov(formula = antler.mass ~ diet, data = df.mass)  
##   
## Terms:  
## diet Residuals  
## Sum of Squares 171543.0 575719.4  
## Deg. of Freedom 2 27  
##   
## Residual standard error: 146.0238  
## Estimated effects may be unbalanced

mass.tuk

## Tukey multiple comparisons of means  
## 95% family-wise confidence level  
##   
## Fit: aov(formula = antler.mass ~ diet, data = df.mass)  
##   
## $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

beam.anova.table

## Analysis of Variance Table  
##   
## Model 1: antler.beam ~ 1  
## Model 2: antler.beam ~ diet  
## Res.Df RSS Df Sum of Sq F Pr(>F)  
## 1 29 108585   
## 2 27 99981 2 8603.7 1.1617 0.3281

beam.pairwise.p

##   
## Pairwise comparisons using t tests with pooled SD   
##   
## data: df.beam$antler.beam and df.beam$diet   
##   
## Hi.Hi Hi.Lo  
## Hi.Lo 0.17 -   
## Lo.Hi 0.83 0.24   
##   
## P value adjustment method: none

beam.aov

## Call:  
## aov(formula = antler.beam ~ diet, data = df.beam)  
##   
## Terms:  
## diet Residuals  
## Sum of Squares 8603.74 99981.22  
## Deg. of Freedom 2 27  
##   
## Residual standard error: 60.85235  
## Estimated effects may be unbalanced

beam.tuk

## Tukey multiple comparisons of means  
## 95% family-wise confidence level  
##   
## Fit: aov(formula = antler.beam ~ diet, data = df.beam)  
##   
## $diet  
## diff lwr upr p adj  
## Hi.Lo-Hi.Hi 38.50262 -28.97224 105.97748 0.3476276  
## Lo.Hi-Hi.Hi 5.88262 -61.59224 73.35748 0.9745943  
## Lo.Hi-Hi.Lo -32.62000 -100.09486 34.85486 0.4641737

circum.anova.table

## Analysis of Variance Table  
##   
## Model 1: antler.circum ~ 1  
## Model 2: antler.circum ~ diet  
## Res.Df RSS Df Sum of Sq F Pr(>F)  
## 1 29 12338   
## 2 27 10877 2 1460.5 1.8127 0.1825

circum.pairwise.p

##   
## Pairwise comparisons using t tests with pooled SD   
##   
## data: df.circum$antler.circum and df.circum$diet   
##   
## Hi.Hi Hi.Lo  
## Hi.Lo 0.157 -   
## Lo.Hi 0.084 0.739  
##   
## P value adjustment method: none

circum.aov

## Call:  
## aov(formula = antler.circum ~ diet, data = df.circum)  
##   
## Terms:  
## diet Residuals  
## Sum of Squares 1460.498 10877.047  
## Deg. of Freedom 2 27  
##   
## Residual standard error: 20.07121  
## Estimated effects may be unbalanced

circum.tuk

## Tukey multiple comparisons of means  
## 95% family-wise confidence level  
##   
## Fit: aov(formula = antler.circum ~ diet, data = df.circum)  
##   
## $diet  
## diff lwr upr p adj  
## Hi.Lo-Hi.Hi 13.058302 -9.197247 35.31385 0.3280747  
## Lo.Hi-Hi.Hi 16.078251 -6.177298 38.33380 0.1915525  
## Lo.Hi-Hi.Lo 3.019949 -19.235600 25.27550 0.9396505

spread.anova.table

## Analysis of Variance Table  
##   
## Model 1: antler.spread ~ 1  
## Model 2: antler.spread ~ diet  
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 29 96798   
## 2 27 58919 2 37879 8.6793 0.001228 \*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

spread.pairwise.p

##   
## Pairwise comparisons using t tests with pooled SD   
##   
## data: df.spread$antler.spread and df.spread$diet   
##   
## Hi.Hi Hi.Lo   
## Hi.Lo 0.00036 -   
## Lo.Hi 0.00998 0.20172  
##   
## P value adjustment method: none

spread.aov

## Call:  
## aov(formula = antler.spread ~ diet, data = df.spread)  
##   
## Terms:  
## diet Residuals  
## Sum of Squares 37879.46 58918.87  
## Deg. of Freedom 2 27  
##   
## Residual standard error: 46.71381  
## Estimated effects may be unbalanced

spread.tuk

## Tukey multiple comparisons of means  
## 95% family-wise confidence level  
##   
## Fit: aov(formula = antler.spread ~ diet, data = df.spread)  
##   
## $diet  
## diff lwr upr p adj  
## Hi.Lo-Hi.Hi 85.23267 33.435024 137.0303 0.0010122  
## Lo.Hi-Hi.Hi 57.89582 6.098174 109.6935 0.0261561  
## Lo.Hi-Hi.Lo -27.33685 -79.134496 24.4608 0.4027839