Homework Assignment 4

Brendan Smith

March 3, 2016

Objective Statement: The prupose of this lab is to determine if the linear models we are developing can adequately characterize the biomass found in the studied habitats. Further, we must compare sites and genera, determining which vary more than others. We are building upon previous labs, and developing a full linear model that relates tree height as a function of DBH and site or genus. A new report will be created that shows the model results for the height as a function of DBH and a factor, being site or genus.

Methods: We begin by importing the "cleaned" riparian data frame from the previous homework. We are to first detemine if there is a systematic bias in the heigh variation by site, then to use genus as a desired factor. The systematic bias is studied by creating a summary subset and analyzing.

Data: The data utilized in this lab is taken from the previous two labs. These data are of several genera of trees' diameter at breast height (DBH) and height, which has been reduced to be of only five most common genera found in the original dataset. Additionally, the height was measured in meters and has been scaled to centimeters for data analysis purposes. DBH was measured in centimeters.

Code: The new code introduced and used in this assignment are `lapply`, `for` loop, `do.call()`, `with()`, `aov()` and `TukeyHSD()`.

Results: We begin our analysis as usual by importing the dataset from the comma separated value file:

```r
# Load libraries
library(stats)
library(HH)
```

```
## Loading required package: lattice

## Loading required package: grid

## Loading required package: latticeExtra

## Loading required package: RColorBrewer

## Loading required package: multcomp

## Loading required package: mvtnorm

## Loading required package: survival

## Loading required package: TH.data

## Loading required package: MASS

##
## Attaching package: 'TH.data'
## The following object is masked from 'package:MASS':
##
##     geyser
```

```
## Loading required package: gridExtra
```

```r
# Load Ripdata and place into a dataframe
rip <- read.csv("newripdata_survey.csv",sep = ",",header = TRUE)
# Add an object that scales the value of height from meters to centimeters
rip$htcm <- rip$Woody_Height_m*100
```

We proceed to make necessary adjustments to the dataframe for data manipulation purposes:

```r
#Concatenate ProjCode and Plot.Name using the paste() function, typecast as a factor, then place these
rip$projplot <- as.factor(paste(rip$ProjCode,rip$Plot.Name))

#use tapply() to cycle through each project plot and generate stats
#where 'htcm' is height in cm
ripsum <- data.frame(cbind(tapply(rip$htcm,rip$projplot,mean),tapply(rip$htcm,rip$projplot,sd),tapply(r
#add column names
#(height mean, height standard deviation, number of plots)
colnames(ripsum) <- c("htcmmn","htcmsd","plot.n")
#add a projplot column (from row names) to ripsum
ripsum$projplot <- as.factor(rownames(ripsum))
```

```r
#subset for plots with more than one measurement
ripsum <- ripsum[ripsum$plot.n > 1,]
#Add a proj column and populate with the first five letters of projplot (the ProjCode) via the substr()
ripsum$proj <- as.factor(substr(ripsum$projplot,1,5))
```

We compare the `for` loop operation to the list apply (`lapply`) in order to demonstrate that although the functions have similar outcomes, the list apply is more efficient for evaluating an array of values simultaneously.

```r
#create list of project sites
projlevels <- levels(ripsum$proj) #compare a 'for' loop of summary
for (p in 1:length(projlevels)) print(summary(ripsum[ripsum$proj == projlevels[p],]))
```

```
##      htcmmn          htcmsd            plot.n          projplot
##  Min.   : 204.1   Min.   :  42.43   Min.   :  2.00   COSRP 1: 1
##  1st Qu.: 769.0   1st Qu.: 277.50   1st Qu.:  9.50   COSRP 2: 1
##  Median :1000.0   Median : 436.05   Median : 18.00   COSRP 3: 1
##  Mean   :1042.3   Mean   : 459.94   Mean   : 25.23   COSRP 4: 1
##  3rd Qu.:1269.2   3rd Qu.: 624.36   3rd Qu.: 32.50   COSRP 5: 1
##  Max.   :3330.0   Max.   :1031.60   Max.   :111.00   COSRP 6: 1
##                                                      (Other):81
##     proj
##  COSRP:87
##  HEROW: 0
##  NAPSO: 0
##  SACTO: 0
##
##
```

```
##
##      htcmmn           htcmsd             plot.n              projplot
##  Min.   : 421.3   Min.   :  68.18   Min.   : 8.00   HEROW RIP01:1
##  1st Qu.: 505.6   1st Qu.: 141.83   1st Qu.:11.50   HEROW RIP02:1
##  Median : 883.0   Median : 218.21   Median :27.00   HEROW RIP03:1
##  Mean   : 978.1   Mean   : 401.83   Mean   :39.14   HEROW RIP04:1
##  3rd Qu.:1332.6   3rd Qu.: 546.12   3rd Qu.:62.00   HEROW RIP05:1
##  Max.   :1866.0   Max.   :1150.50   Max.   :92.00   HEROW RIP06:1
##                                                     (Other)    :1
##     proj
##  COSRP:0
##  HEROW:7
##  NAPSO:0
##  SACTO:0
##
##
##
##      htcmmn           htcmsd           plot.n                projplot
##  Min.   : 671.0   Min.   :171.6   Min.   : 4.00   NAPSO Crp2013_509:1
##  1st Qu.: 745.4   1st Qu.:472.4   1st Qu.: 6.75   NAPSO SO2013_200 :1
##  Median : 814.6   Median :556.3   Median :17.50   NAPSO SO2013_202 :1
##  Mean   : 977.8   Mean   :538.2   Mean   :24.00   NAPSO SO2013_203 :1
##  3rd Qu.:1016.4   3rd Qu.:649.9   3rd Qu.:29.25   NAPSO SO2013_207 :1
##  Max.   :1888.3   Max.   :819.5   Max.   :81.00   NAPSO SO2013_211 :1
##                                                   (Other)          :2
##     proj
##  COSRP:0
##  HEROW:0
##  NAPSO:8
##  SACTO:0
##
##
##
##      htcmmn           htcmsd             plot.n                projplot
##  Min.   : 443.2   Min.   :  93.04   Min.   : 2.00   SACTO EW2013_100: 1
##  1st Qu.: 678.7   1st Qu.: 254.57   1st Qu.: 4.00   SACTO EW2013_101: 1
##  Median : 977.0   Median : 584.23   Median : 7.00   SACTO EW2013_102: 1
##  Mean   :1071.0   Mean   : 623.96   Mean   :11.82   SACTO EW2013_103: 1
##  3rd Qu.:1256.6   3rd Qu.: 882.65   3rd Qu.:17.00   SACTO EW2013_106: 1
##  Max.   :2532.9   Max.   :1668.77   Max.   :59.00   SACTO EW2013_110: 1
##                                                     (Other)         :38
##     proj
##  COSRP: 0
##  HEROW: 0
##  NAPSO: 0
##  SACTO:44
##
##
##
```

```r
#with a summary using lapply() (known as list apply)
lapply(projlevels, function(x) summary(ripsum[ripsum$proj == x,]))
```

```
## [[1]]
##      htcmmn           htcmsd            plot.n          projplot 
##  Min.   : 204.1   Min.   :  42.43   Min.   :  2.00   COSRP 1: 1  
##  1st Qu.: 769.0   1st Qu.: 277.50   1st Qu.:  9.50   COSRP 2: 1  
##  Median :1000.0   Median : 436.05   Median : 18.00   COSRP 3: 1  
##  Mean   :1042.3   Mean   : 459.94   Mean   : 25.23   COSRP 4: 1  
##  3rd Qu.:1269.2   3rd Qu.: 624.36   3rd Qu.: 32.50   COSRP 5: 1  
##  Max.   :3330.0   Max.   :1031.60   Max.   :111.00   COSRP 6: 1  
##                                                      (Other):81  
##     proj   
##  COSRP:87  
##  HEROW: 0  
##  NAPSO: 0  
##  SACTO: 0  
##            
##            
##            
## 
## [[2]]
##      htcmmn           htcmsd            plot.n            projplot
##  Min.   : 421.3   Min.   :  68.18   Min.   : 8.00   HEROW RIP01:1  
##  1st Qu.: 505.6   1st Qu.: 141.83   1st Qu.:11.50   HEROW RIP02:1  
##  Median : 883.0   Median : 218.21   Median :27.00   HEROW RIP03:1  
##  Mean   : 978.1   Mean   : 401.83   Mean   :39.14   HEROW RIP04:1  
##  3rd Qu.:1332.6   3rd Qu.: 546.12   3rd Qu.:62.00   HEROW RIP05:1  
##  Max.   :1866.0   Max.   :1150.50   Max.   :92.00   HEROW RIP06:1  
##                                                     (Other)    :1  
##     proj  
##  COSRP:0  
##  HEROW:7  
##  NAPSO:0  
##  SACTO:0  
##           
##           
##           
## 
## [[3]]
##      htcmmn           htcmsd          plot.n                  projplot
##  Min.   : 671.0   Min.   :171.6   Min.   : 4.00   NAPSO Crp2013_509:1  
##  1st Qu.: 745.4   1st Qu.:472.4   1st Qu.: 6.75   NAPSO S02013_200 :1  
##  Median : 814.6   Median :556.3   Median :17.50   NAPSO S02013_202 :1  
##  Mean   : 977.8   Mean   :538.2   Mean   :24.00   NAPSO S02013_203 :1  
##  3rd Qu.:1016.4   3rd Qu.:649.9   3rd Qu.:29.25   NAPSO S02013_207 :1  
##  Max.   :1888.3   Max.   :819.5   Max.   :81.00   NAPSO S02013_211 :1  
##                                                   (Other)          :2  
##     proj  
##  COSRP:0  
##  HEROW:0  
##  NAPSO:8  
##  SACTO:0  
##           
##           
##           
## 
## [[4]]
##      htcmmn           htcmsd            plot.n                   projplot
##  Min.   : 443.2   Min.   :  93.04   Min.   : 2.00   SACTO EW2013_100: 1
##  1st Qu.: 678.7   1st Qu.: 254.57   1st Qu.: 4.00   SACTO EW2013_101: 1
##  Median : 977.0   Median : 584.23   Median : 7.00   SACTO EW2013_102: 1
##  Mean   :1071.0   Mean   : 623.96   Mean   :11.82   SACTO EW2013_103: 1
##  3rd Qu.:1256.6   3rd Qu.: 882.65   3rd Qu.:17.00   SACTO EW2013_106: 1
##  Max.   :2532.9   Max.   :1668.77   Max.   :59.00   SACTO EW2013_110: 1
##                                                     (Other)         :38
##     proj
##  COSRP: 0
##  HEROW: 0
##  NAPSO: 0
##  SACTO:44
##
##
##
```

It can be seen that while the outcome is the same, the setup and application of the for loop is somewhat inefficient in that we must manually indicate the start and stop indeces in order to iterate through the entire array/vector individually with the `for` loop, whereas with the list apply, all elements are evaluated by the function automatically.

We then use `lapply` to randomly select six sample plot summaries from each project site. This is done by by first introducing a variable that is set to the integer 6, the number of samples desired. The function `sample` is then used to output the desired number of samples (6) randomly from each project site. To execute this for all project sites, the `lapply` function is utilized. These values are stored and combined by row using the `rbind()` function along with `do.call()`. Finally, the summary is output.

```r
nsamples <- 6 #Set the number of samples
ripres <- lapply(projlevels, function(x) ripsum[which(ripsum$proj == x),][sample(nrow(ripsum[which(rips
# combine samples by row using rbind()
# and by calling ripres lapply function from do.call()
ripsample <- do.call(rbind,ripres)
summary(ripsample$proj)
```

```
## COSRP HEROW NAPSO SACTO
##     6     6     6     6
```

The coefficient of variation (CV) then added to the summary table by calculating the CV by means of the `with()` function. The `with()` function evaluates an `R` expression (second input term) of the input data (first term). In this case, we are evaluating the coefficient of variation, which is the standard deviation divided by the mean. This value is then stored into our dataframe.

```r
#calculate CV using with(data,calc)
ripsample$cv <- with(ripsample, htcmsd / htcmmn)
```

Equipped with the coefficient of variation, we can now run a one-way analysis of variation (ANOVA) on the data frame. This is done by utilizing the `aov` function, which we input the CV as a function of the project code. The output is the sum of squares, degrees of freedom and the residual standard error, all of which are stored in a new variable followed by a summary output.

```
rip.proj.cv.aov = aov(cv~proj,data=ripsample)
summary(rip.proj.cv.aov)
```

```
##             Df Sum Sq Mean Sq F value Pr(>F)
## proj         3 0.1217 0.04057   0.768  0.525
## Residuals   20 1.0561 0.05281
```

```
#compare it against
summary.lm(rip.proj.cv.aov)
```

```
##
## Call:
## aov(formula = cv ~ proj, data = ripsample)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.40388 -0.20518  0.04908  0.15089  0.41258
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.49757    0.09381   5.304 3.43e-05 ***
## projHEROW   -0.12366    0.13267  -0.932    0.362
## projNAPSO    0.06055    0.13267   0.456    0.653
## projSACTO    0.03846    0.13267   0.290    0.775
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2298 on 20 degrees of freedom
## Multiple R-squared:  0.1033, Adjusted R-squared:  -0.03116
## F-statistic: 0.7684 on 3 and 20 DF,  p-value: 0.5252
```

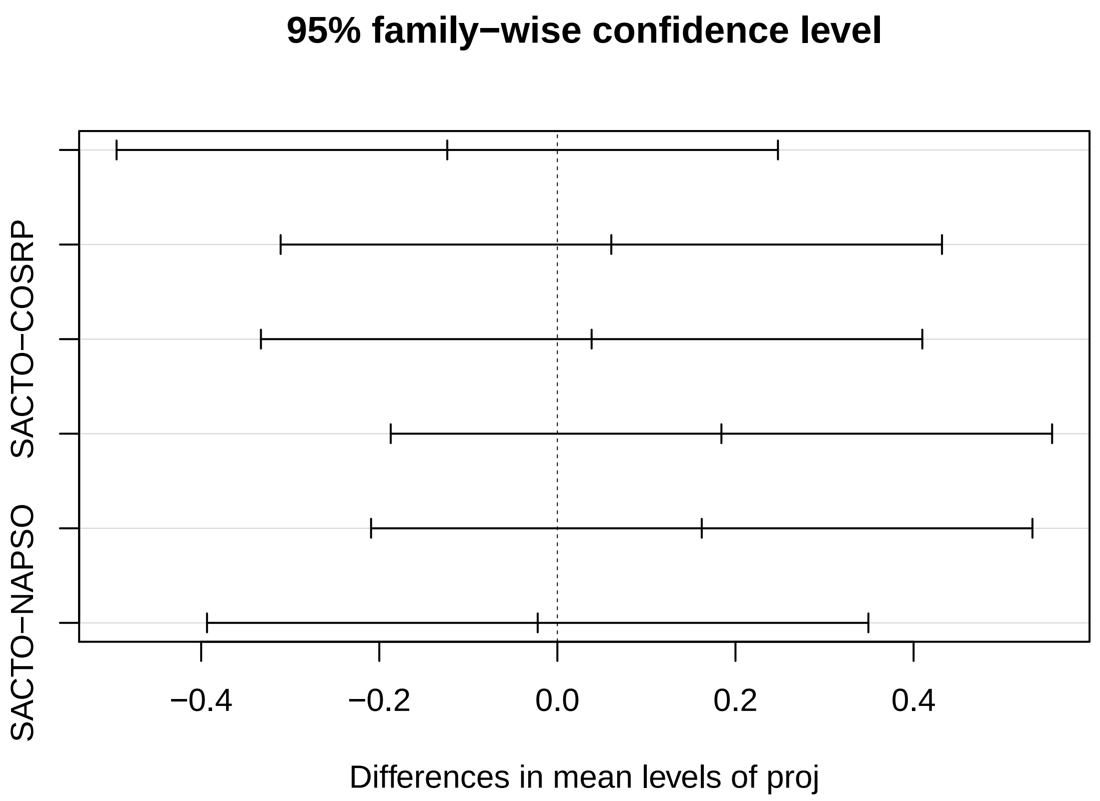
The main differences between the summary of the ANOVA results and the summary of the linear model of the ANOVA is that the lm version yields information regarding the individual project sites and the residuals, whereas the summary of the ANOVA yields information regarding all project sites and residuals, limited to DOF, sum of squares, mean squared, F value and probability.

A Tukey test is performed to check for significant differences between sites, and print out the results. We can see from the print out of the Tukey test and the plot that there is not a significant difference between sites.

```
rip.aov.hsd <- TukeyHSD(rip.proj.cv.aov)
rip.aov.hsd
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = cv ~ proj, data = ripsample)
##
## $proj
##                    diff        lwr       upr     p adj
## HEROW-COSRP -0.12366235 -0.4950014 0.2476767 0.7881577
## NAPSO-COSRP  0.06055130 -0.3107877 0.4318903 0.9675923
## SACTO-COSRP  0.03846479 -0.3328742 0.4098038 0.9912446
## NAPSO-HEROW  0.18421365 -0.1871254 0.5555527 0.5205912
## SACTO-HEROW  0.16212714 -0.2092119 0.5334661 0.6206052
## SACTO-NAPSO -0.02208651 -0.3934255 0.3492525 0.9983020
```

```
plot(rip.aov.hsd)
```



In step two we are evaluating the analysis of covariance (ANCOVA), in which we are using the height as a funtion of DBH and genus as a factor. We begin by creating a general linear model where the height of the tree is a function of the DBH and a function of genus separately.

```
# Create a linear model for height versus DBH
rip.cov.htdbh <-glm(rip$htcm~rip$Woody_DBH_cm)
# Generate summary for this model
summary(rip.cov.htdbh)
```

```
##
## Call:
## glm(formula = rip$htcm ~ rip$Woody_DBH_cm)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3425.7   -260.8    -71.8    198.0   9815.7
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       555.214     10.546   52.64   <2e-16 ***
## rip$Woody_DBH_cm   20.786      0.387   53.71   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 197786.6)
##
##     Null deviance: 1201446793  on 3191  degrees of freedom
## Residual deviance:  630939390  on 3190  degrees of freedom
## AIC: 47989
##
## Number of Fisher Scoring iterations: 2
```

```
summary.lm(rip.cov.htdbh)
```

```
##
## Call:
## glm(formula = rip$htcm ~ rip$Woody_DBH_cm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3425.7  -260.8   -71.8   198.0  9815.7
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       555.214     10.546   52.64   <2e-16 ***
## rip$Woody_DBH_cm   20.786      0.387   53.71   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 444.7 on 3190 degrees of freedom
## Multiple R-squared:  0.4749, Adjusted R-squared:  0.4747
## F-statistic:  2884 on 1 and 3190 DF,  p-value: < 2.2e-16
```

```
# Create a linear model for height versus Genus
rip.cov.htg <-glm(rip$htcm~rip$Genus)
# Generate summary for this model
summary(rip.cov.htg)
```

```
##
## Call:
## glm(formula = rip$htcm ~ rip$Genus)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1264.3   -323.4    -85.1    212.7   9789.5
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         707.32      25.47  27.773  < 2e-16 ***
## rip$GenusFraxinus   -96.81      35.76  -2.707  0.00683 **
## rip$GenusPopulus    676.99      31.69  21.361  < 2e-16 ***
## rip$GenusQuercus    258.19      32.85   7.860 5.22e-15 ***
## rip$GenusSalix       37.82      32.08   1.179  0.23859
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for gaussian family taken to be 293171.6)
##
##     Null deviance: 1201446793  on 3191  degrees of freedom
## Residual deviance:  934337819  on 3187  degrees of freedom
## AIC: 49248
##
## Number of Fisher Scoring iterations: 2
```

```
summary.lm(rip.cov.htg)
```

```
##
## Call:
## glm(formula = rip$htcm ~ rip$Genus)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1264.3  -323.4   -85.1   212.7  9789.5
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)           707.32      25.47  27.773  < 2e-16 ***
## rip$GenusFraxinus     -96.81      35.76  -2.707  0.00683 **
## rip$GenusPopulus      676.99      31.69  21.361  < 2e-16 ***
## rip$GenusQuercus      258.19      32.85   7.860 5.22e-15 ***
## rip$GenusSalix         37.82      32.08   1.179  0.23859
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 541.5 on 3187 degrees of freedom
## Multiple R-squared:  0.2223, Adjusted R-squared:  0.2213
## F-statistic: 227.8 on 4 and 3187 DF,  p-value: < 2.2e-16
```

```
# Create a linear model for height as a function of DBH and Genus
rip.cov.htdbhg <-glm(rip$htcm~rip$Woody_DBH_cm*rip$Genus)
# Generate summary for this model
summary(rip.cov.htdbhg)
```

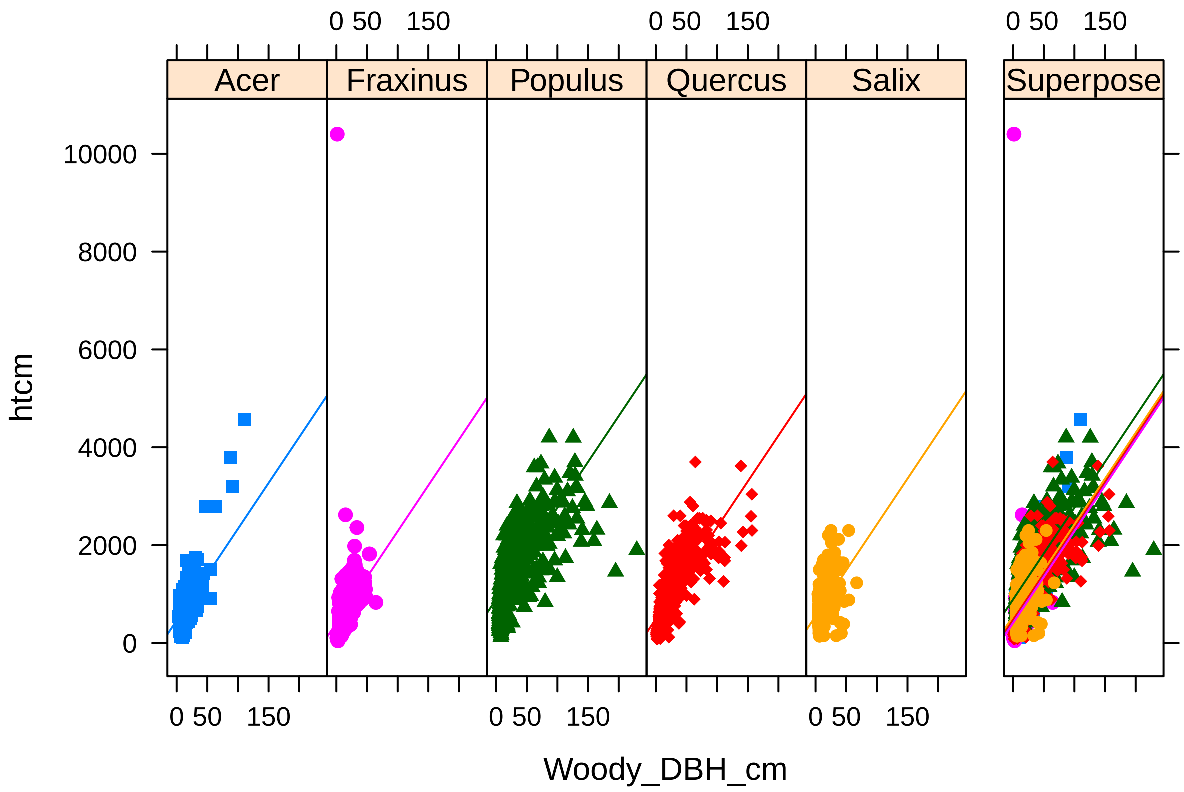
```
##
## Call:
## glm(formula = rip$htcm ~ rip$Woody_DBH_cm * rip$Genus)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2721.1   -231.5    -25.1    193.9  10030.7
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         303.993     29.714  10.231  < 2e-16
## rip$Woody_DBH_cm                     30.549      1.735  17.605  < 2e-16
## rip$GenusFraxinus                    28.923     41.305   0.700 0.483838
## rip$GenusPopulus                    661.299     35.487  18.635  < 2e-16
## rip$GenusQuercus                    143.344     37.167   3.857 0.000117
## rip$GenusSalix                      210.982     39.513   5.340 9.97e-08
## rip$Woody_DBH_cm:rip$GenusFraxinus   -4.592      2.677  -1.715 0.086381
## rip$Woody_DBH_cm:rip$GenusPopulus   -14.620      1.809  -8.083 8.86e-16
## rip$Woody_DBH_cm:rip$GenusQuercus   -10.035      1.849  -5.427 6.17e-08
## rip$Woody_DBH_cm:rip$GenusSalix      -8.593      2.696  -3.187 0.001452
##
## (Intercept)                        ***
## rip$Woody_DBH_cm                   ***
## rip$GenusFraxinus
## rip$GenusPopulus                   ***
## rip$GenusQuercus                   ***
## rip$GenusSalix                     ***
## rip$Woody_DBH_cm:rip$GenusFraxinus .
## rip$Woody_DBH_cm:rip$GenusPopulus  ***
## rip$Woody_DBH_cm:rip$GenusQuercus  ***
## rip$Woody_DBH_cm:rip$GenusSalix    **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 161859.5)
##
##     Null deviance: 1201446793  on 3191  degrees of freedom
## Residual deviance:  515037025  on 3182  degrees of freedom
## AIC: 47357
##
## Number of Fisher Scoring iterations: 2
```

```r
summary.lm(rip.cov.htdbhg)
```

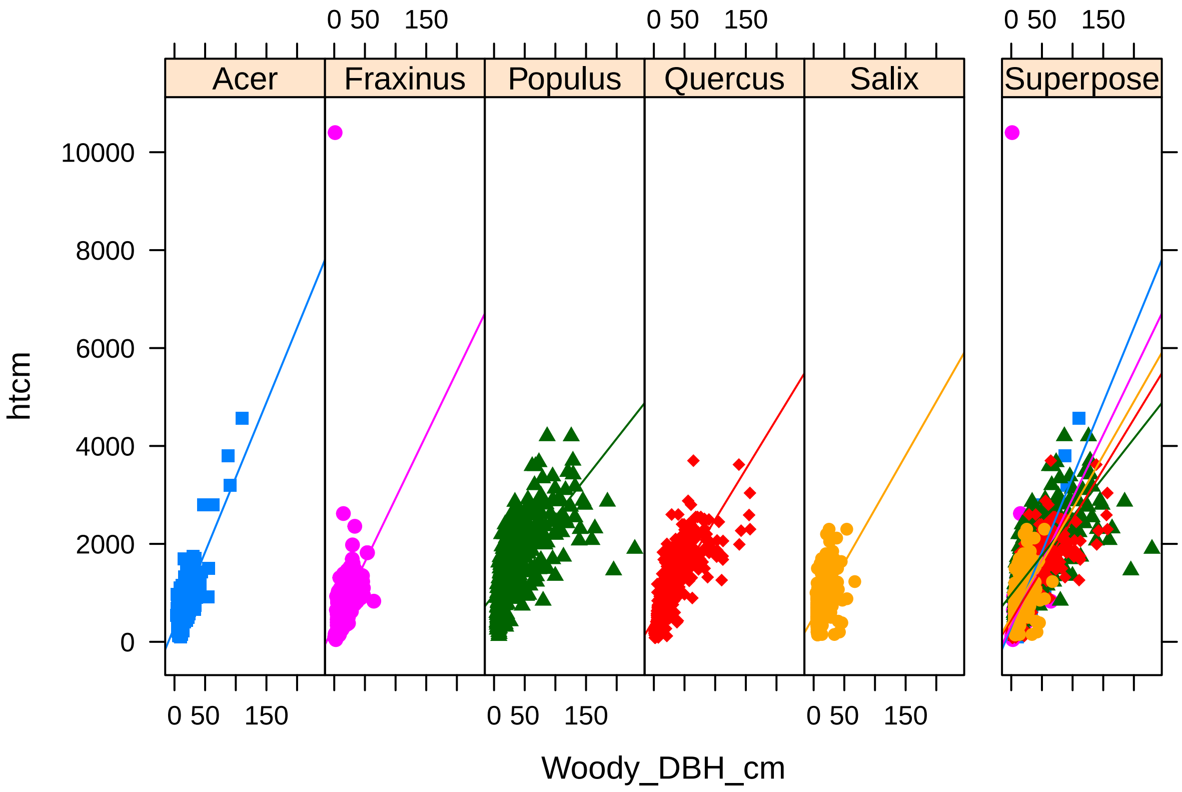
```
##
## Call:
## glm(formula = rip$htcm ~ rip$Woody_DBH_cm * rip$Genus)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2721.1  -231.5   -25.1   193.9 10030.7
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         303.993     29.714  10.231  < 2e-16
## rip$Woody_DBH_cm                     30.549      1.735  17.605  < 2e-16
## rip$GenusFraxinus                    28.923     41.305   0.700 0.483838
## rip$GenusPopulus                    661.299     35.487  18.635  < 2e-16
## rip$GenusQuercus                    143.344     37.167   3.857 0.000117
## rip$GenusSalix                      210.982     39.513   5.340 9.97e-08
## rip$Woody_DBH_cm:rip$GenusFraxinus   -4.592      2.677  -1.715 0.086381
## rip$Woody_DBH_cm:rip$GenusPopulus   -14.620      1.809  -8.083 8.86e-16
## rip$Woody_DBH_cm:rip$GenusQuercus   -10.035      1.849  -5.427 6.17e-08
## rip$Woody_DBH_cm:rip$GenusSalix      -8.593      2.696  -3.187 0.001452
##
## (Intercept)                        ***
## rip$Woody_DBH_cm                   ***
## rip$GenusFraxinus
## rip$GenusPopulus                   ***
## rip$GenusQuercus                   ***
## rip$GenusSalix                     ***
## rip$Woody_DBH_cm:rip$GenusFraxinus .
## rip$Woody_DBH_cm:rip$GenusPopulus  ***
## rip$Woody_DBH_cm:rip$GenusQuercus  ***
## rip$Woody_DBH_cm:rip$GenusSalix    **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 402.3 on 3182 degrees of freedom
## Multiple R-squared:  0.5713, Adjusted R-squared:  0.5701
## F-statistic: 471.2 on 9 and 3182 DF,  p-value: < 2.2e-16
```

The next step is to generate an `ancovaplot()` for the two model formulations, with and without the interaction. We first plot the height as a function of the DBH and the factor Genus:

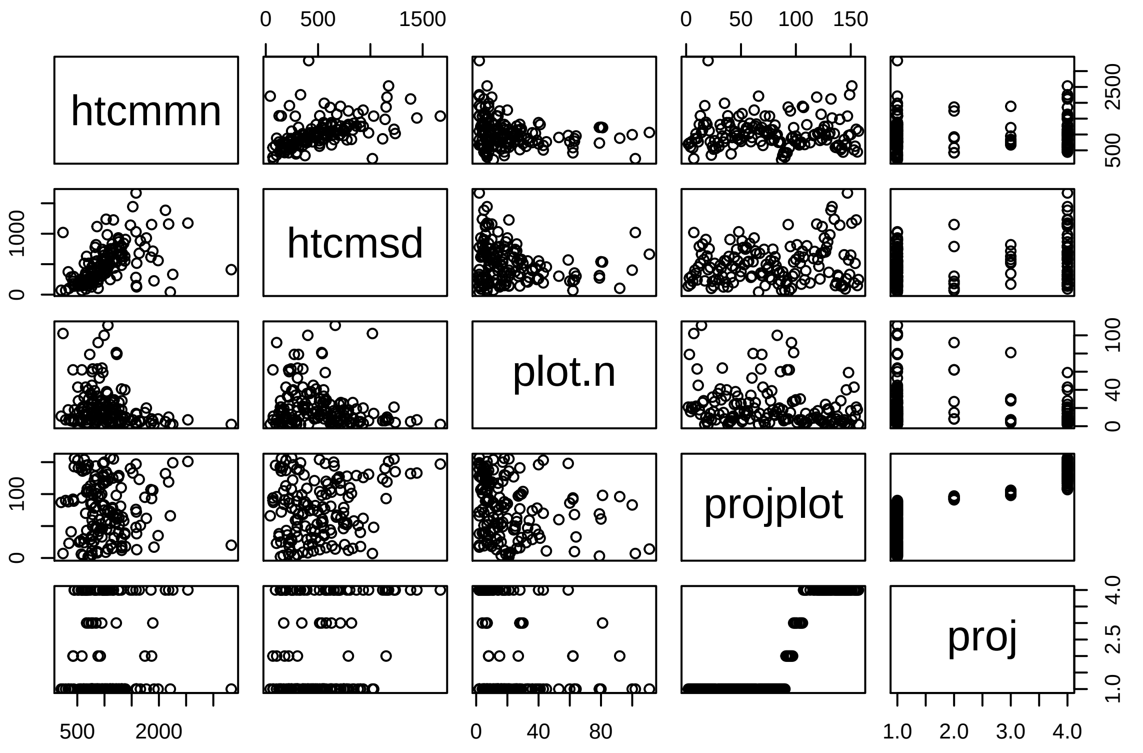
```r
ancovaplot(htcm~Woody_DBH_cm+Genus,data=rip)
```



```r
ancovaplot(htcm~Woody_DBH_cm*Genus,data=rip)
```



```
plot(ripsum)
```



Discussion: Through the use of the coefficient of variation, we were able to run a one-way ANOVA, followed by checking for significant differences between sites using the Tukey test. By analyzing the summary and plot of these results, we can see that there is not significant differences in the variation of the height at these project codes. This is to say that height variability does not differ much between project codes. By utilizing the Tukey test, we are analyzing the differences between the means of the levels of the factor created by the one-way ANOVA. We can see that the difference between the means are relatively low, indicating that there is not a significant difference. Furthermore, if we look at the plot generated utilizing the Tukey data, we can see that all the project codes' differences in mean levels fall around zero, reinforcing our notion.

Limitations: A limitaiton that was encountered in this homework assignment was the learning how to use the `ancovaplot()` function properly. Though the input parameters seemed to be accurate, an error was always thrown until I realized that you must indicate the dataframe objects and then identify the dataframe individually.