B.J. Enquist
Scaling In Biology
Intro to R and Regression tutorial

Download R Studio

A quick R Tutorial – importing data, plotting it, basic manipulations, regression and alternative regression models.

In this tutorial I am assuming you have access to two files (bigtrees.cvs and slopes.R). Both files are available at our GitHub class site https://github.com/benquist/ECOL586

Note, to use slopes.R there cannot be missing values. Let’s import a file that contains the diameters and heights of the biggest trees in the USA. The file is named bigtrees.cvs however, we need to import the file into R. We use the read.table command. Make sure you know where the file is located on your hard drive.

```
> bigtrees <- read.table("/Users/benquist/Desktop/bigtrees.csv",sep=",",header=T)
```

Let’s look at the imported data

```
> bigtrees
                    SpecName Circumference.inches Height.feet Spread.feet Diameter..in.    X
1         Acacia rigidula                16         26         29         5.09 3.14
2         Ailanthus altissima               248         67         64        78.94  NA
3          Chamaecyparis nootkatensis              451        124         27       143.56   NA
```

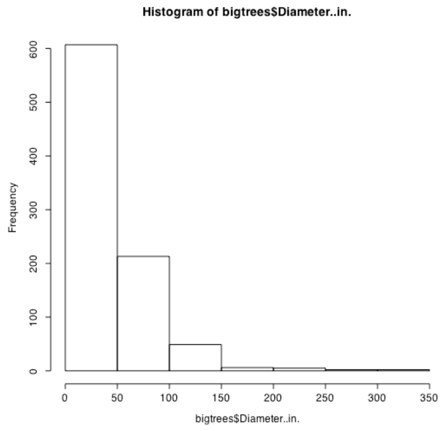
etc. etc.

What are the data column names of the file bigrees?

```
> names(bigtrees)
[1] "SpecName"           "Circumference.inches" "Height.feet"         "Spread.feet"
[5] "Diameter..in."       "X"                     "Height.in."          "Spread..in."
```

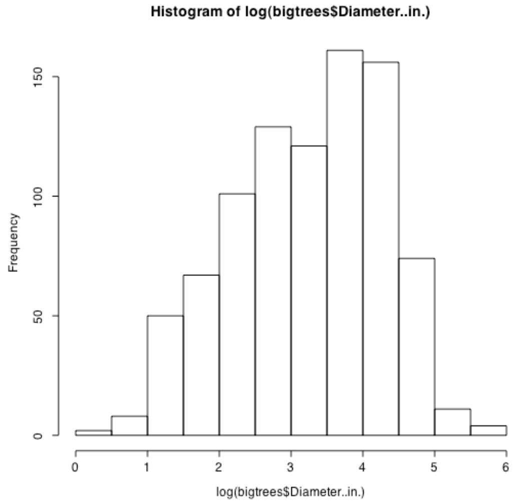
Uggh, the data are not metric and are in imperial units. Let’s look at summarized data. Plot a histogram of one of the columns.

```
> hist(bigtrees $Diameter..in.)
```



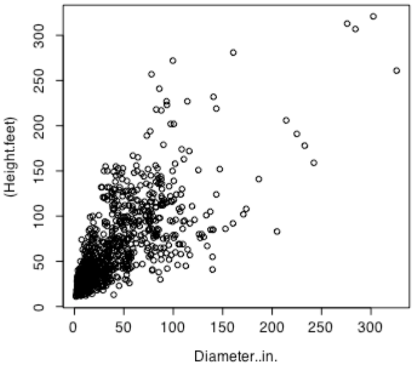
Now, let’s log transform the column. Note in R the ‘log’ command gives you the natural log (e) transformation while ‘log10’ gives $\log_{10}$ transformation.

```
> hist(log(bigtrees $Diameter..in.))
```



Plotting the data . . .

```
> plot((Height.feet)~(Diameter..in.),bigtrees)
```



Now, let's log transform the axes . .

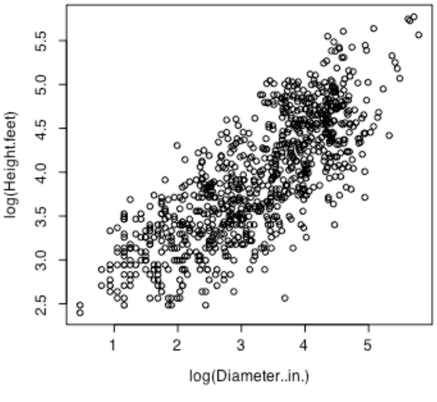
```
> plot(log(Height.feet)~log(Diameter.in),bigtrees)
Error in log(Diameter.in) : Object "Diameter.in" not found
```

Opps!

```
> plot(log(Height.feet)~log(Diameter..in),bigtrees)
Error in log(Diameter..in) : Object "Diameter..in" not found
```

Try it again . . .

```
> plot(log(Height.feet)~log(Diameter..in.),bigtrees)
```



Let's fit an OLS regression through the log transformed data . . . use the lm command

```
> summary(lm(log(Height.feet)~log(Diameter..in.),bigtrees))

Call:
```

```
lm(formula = log(Height.feet) ~ log(Diameter..in.), data = bigtrees)

Residuals:
     Min       1Q   Median       3Q      Max 
-1.59058 -0.31301  0.01337  0.29303  1.04865 

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)    
(Intercept)         2.26770    0.04696   48.29   <2e-16 ***
log(Diameter..in.)  0.51250    0.01376   37.26   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 

Residual standard error: 0.4296 on 882 degrees of freedom
Multiple R-Squared: 0.6115,  Adjusted R-squared: 0.611 
F-statistic:  1388 on 1 and 882 DF,  p-value: < 2.2e-16
```

Above the coefficients are the Intercept as labeled as well as log(Diameter..in.) **which is the slope. Note, you can use the Std. Error to calc. the 95% Cis.**

```
> plot((log(Height.feet)~(log(Diameter..in.)),bigtrees)
Error: syntax error
```

Opps! sloppy yet again . .

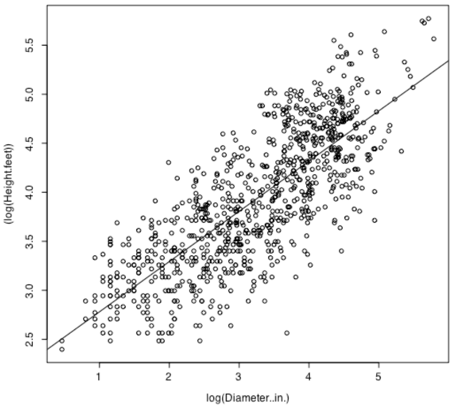
```
> plot((log(Height.feet))~(log(Diameter..in.)),bigtrees)
```

Let’s plot the regression line on top of the data plot. First, generate the line to plot . . .

```
> lm.HD_Line <-lm(lm((log(Height.feet))~(log(Diameter..in.)),bigtrees))
```

Now, plot it on the graph

```
> abline(lm.HD_Line)
```

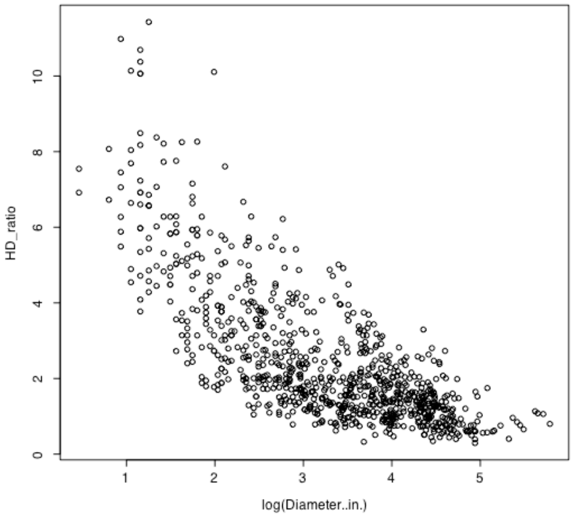


Let's manipulate the data. Create a new variable - the ratio of height to diameter (the HD_ratio).

```
> HD_ratio <-((bigtrees$Height.feet)/(bigtrees$Diameter..in.))
```

Let's plot out this new variable. . .

```
> plot(HD_ratio~(log(Diameter..in.)), bigtrees)
```

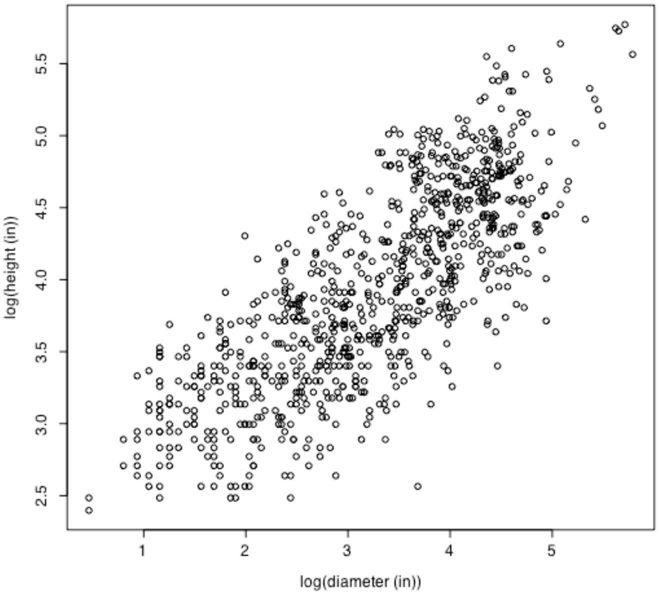


Cool. The H/D ratio decreases with increases in diameter indicating that trees are more 'leggy' at smaller sizes. Notice, the fitted exponent to this relationship would be steeper than –1 indicating that the negative relationship is not due to simply Y now being divided by X.

```
> plot(log(HD_ratio)~(log(Diameter..in.)), bigtrees)
```

We can make the graphs look a little better by adding some x and y axes.

```
>plot(log(bigtrees$Diameter..in.),log(bigtrees$Height.feet),xlab="log(diameter
(in))",ylab="log(height (in))")
```



Let's use the slopes.s routine to calculate the OLS, RMA and OLS/Bisector technique. To do so put the file 'slopes.s' on your desktop. Then use the source command to import the commands.

```
> source("/Users/benquist/Desktop/slopes.R")
```

To look at the slopes.R command type the following. Notice that many of dthe commands in the program are just calculating the normal aspects of regression statistics (means, variance, sum of squares, covariances etc.)

```
> slopes
```

```
function(x,y) {
#COMPUTE AVERAGES AND SUMS
XAVG <- mean(x)
YAVG <- mean(y)
RN <- length(x)
SXX <- sum((x-mean(x))^2)
SYY <- sum((y-mean(y))^2)
SXY <- sum((x-mean(x))*(y-mean(y)))
```

```
#COMPUTE THE SLOPE COEFFICIENTS
B <- 0
B[1] <- SXY/SXX
B[2] <- SYY/SXY
B[3] <- (B[1]*B[2] - 1.0 + sqrt((1.0 + B[1]^2)*(1.0 + B[2]^2)))/(B[1] + B[2])
B[4] <- 0.5*(B[2] - 1.0/B[1] + sign(SXY)*sqrt(4.0 + (B[2] - 1.0/B[1])^2))
B[5] <- sign(SXY)*sqrt(B[1]*B[2])
B[6] <- 0.5*(B[1] + B[2])

#COMPUTE INTERCEPT COEFFICIENTS
A <- 0
for(i in 1:6) {
        A[i] <- YAVG - B[i]*XAVG
}

#PREPARE FOR COMPUTATION OF VARIANCES
x <- x - XAVG
y <- y - YAVG
GAM1 <- B[3]/((B[1] + B[2])*sqrt((1.0 + B[1]^2)*(1.0 + B[2]^2)))
GAM2 <- B[4]/(sqrt(4.0*B[1]^2 + (B[1]*B[2] - 1.0)^2))
SUM1 <- sum((x*(y - B[1]*x))^2)
SUM2 <- sum((y*(y - B[2]*x))^2)
SUM3 <- sum(x*y*(y - B[1]*x)*(y - B[2]*x))
COV <- SUM3/(B[1]*SXX^2)

#COMPUTE VARIANCES OF THE SLOPE COEFFICIENTS
SIGB <- 0
SIGB[1] <- SUM1/(SXX^2)
SIGB[2] <- SUM2/(SXY^2)
SIGB[3] <- (GAM1^2)*(((1.0 + B[2]^2)^2)*SIGB[1] + 2.0*(1.0 + B[1]^2)*(1.0 +
B[2]^2)*COV + ((1.0+B[1]^2)^2)*SIGB[2])
SIGB[4] <- (GAM2^2)*(SIGB[1]/B[1]^2 + 2.0*COV + B[1]^2*SIGB[2])
SIGB[5] <- 0.25*(B[2]*SIGB[1]/B[1] + 2.0*COV + B[1]*SIGB[2]/B[2])
SIGB[6] <- 0.25*(SIGB[1] + 2.0*COV + SIGB[2])

#COMPUTE STANDARD DEVIATIONS OF THE INTERCEPT COEFFICIENTS
SIGA <- 0
SIGA[1] <- sum(((y - B[1]*x)*(1.0 - RN*XAVG*x/SXX))^2)
SIGA[2] <- sum(((y - B[2]*x)*(1.0 - RN*XAVG*y/SXY))^2)
SIGA[3] <- sum(((x*(y - B[1]*x)*(1.0 + B[2]^2)/SXX + y*(y - B[2]*x)*(1.0 +
B[1]^2)/SXY)*GAM1*XAVG*RN - y + B[3]*x)^2)
SIGA[4] <- sum(((x*(y - B[1]*x)/SXX + y*(y -
B[2]*x)*(B[1]^2)/SXY)*GAM2*XAVG*RN/sqrt(B[1]^2) - y + B[4]*x)^2)
SIGA[5] <- sum(((x*(y - B[1]*x)*sqrt(B[2]/B[1])/SXX + y*(y -
B[2]*x)*sqrt(B[1]/B[2])/SXY)*0.5*RN*XAVG - y + B[5]*x)^2)
```

```
SIGA[6] <- sum(((x*(y - B[1]*x)/SXX + y*(y - B[2]*x)/SXY)*0.5*RN*XAVG - y +
B[6]*x)^2)

#generates R object
correlation<-cor(x,y)
parameters <-
matrix(data=NA,nrow=6,ncol=4,dimnames=list(c('OLS(Y/X)','OLS(X/Y)','OLS
BISECTOR','ORTHOGONAL','REDUCED MAJ AXIS','MEAN OLS'),
                          c('A','SD(A)','B','SD(B)')))
parameters[,1] <- A
parameters[,2] <- sqrt(SIGA)/RN
parameters[,3] <- B
parameters[,4] <- sqrt(SIGB)
x <- x + XAVG
y <- y + YAVG
return(x,y,correlation,parameters)
}
>
```

Let's plot the data one more time as we are going to plot some regression functions on top of it.

```
> log_height <-(log(bigtrees$Height.feet))
> log_diamter <-(log(bigtrees$Diameter..in.))
```

We will call btreg the output from the slopes routine. Notice, that when we invoke the slopes command we have to specify the x and y variable. The output, btreg, now contains the output of the slopes command.

```
> btreg<-slopes(log(bigtrees$Diameter..in.),log(bigtrees$Height.feet))
Warning message:
multi-argument returns are deprecated in: return(x, y, correlation, parameters)
```

Ignore those messages for now . . . if you type btreg you then can see the output from the slopes program

```
> btreg

$x
  [1] 1.6272778 4.3686881 4.9667531 3.8592549 2.4932055 2.3513753 1.5623463
4.3565806 3.3985272 2.1138430
 [11] 2.2565412 2.1138430 3.5374750 4.1485170 1.9906103 2.0744290 3.1591258
3.9910190 0.8020016 1.5623463
 [21] 3.1036894 3.4196920 3.6003214 2.4932055 3.0449985 4.0647441 3.1591258
4.5723370 4.0970077 2.4105422
```

```
 [31] 3.1859393 2.8063861 4.0483006 4.5042443 2.1871742 4.4120708 2.8808824
2.0744290 3.9114228 3.4503050
```

and so on . . .

```
[859] 4.770685 4.890349 3.178054 3.496508 3.295837 2.564949 2.564949 2.890372
3.555348 3.465736 3.401197
[870] 4.709530 4.110874 2.995732 4.248495 3.988984 2.639057 2.772589 3.218876
2.890372 3.178054 3.496508
[881] 2.708050 3.258097 3.135494 3.401197

$correlation
[1] 0.7819591

$parameters
                  A      SD(A)        B      SD(B)
OLS(Y/X)         2.267698 0.04124988 0.5124952 0.01247845
OLS(X/Y)         1.209841 0.07440756 0.8381499 0.02093609
OLS BISECTOR     1.778218 0.04567799 0.6631786 0.01287128
ORTHOGONAL       2.022768 0.04947388 0.5878953 0.01469820
REDUCED MAJ AXIS 1.803489 0.04573966 0.6553990 0.01298854
MEAN OLS         1.738770 0.04853435 0.6753225 0.01365805
```

Above you will see the various parameters for the OLS, RMA, and OLS Bisector technique. Notice how variable the slope estimates are for the three regression models. Let’s plot out the various calculated slopes from the models.

```
> >plot(log(bigtrees$Diameter..in.),log(bigtrees$Height.feet),xlab="log(diameter
(in))",ylab="log(height (in))")
```

Let’s put the OLS *(Y |X)* line on the graph

```
> abline(btreg$parameters[1,1],btreg$parameters[1,3],lty=1)
```

Let’s put the OLS *(X |Y)* line on the graph

```
> abline(btreg$parameters[2,1],btreg$parameters[2,3],lty=1)
```

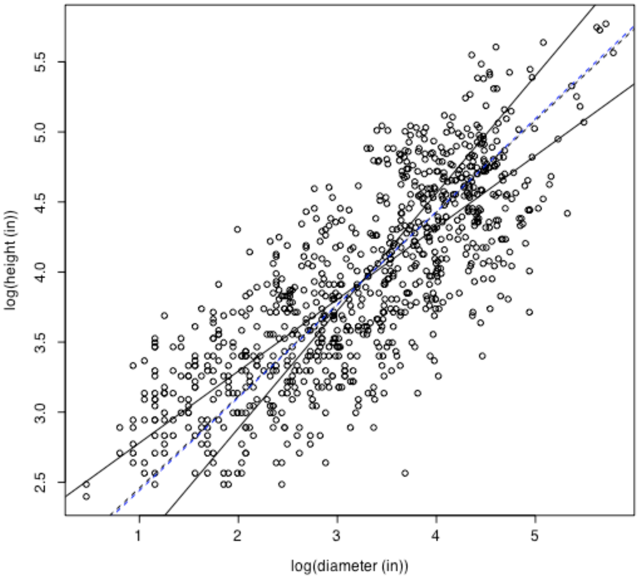
Let’s put the RMA line (dashed line) on the graph

```
> abline(btreg$parameters[5,1],btreg$parameters[5,3],lty=2)
```

Here is the OLS bisector line (dashed green line) on the graph

```
> abline(btreg$parameters[3,1],btreg$parameters[3,3],lty=2, col=4)
```

```
>
```



log(height (in))
2.5
3.0
3.5
4.0
4.5
5.0
5.5
1
2
3
4
5
log(diameter (in))