Lab # 2. Describing and Visualizing Data Part 2 and probability

Melissa Pulley

2024-02-02

You will need to submit the your code. You can answer the questions by annotating your answers in the code, or, if you prefer by uploading a word document. You can also submit an Rmarkdown document

These boxes will inform you of things you need to submit or questions you need to answer!

1 Introduction

Today's lab assignment is an extension of what we did last week, in which, we learnt how to:

1. Enter and save data in MS Excel
2. Import a .csv data table into R using the read.csv() function
3. Extract elements (k) of a vector using [k]
4. Extract rows (i), columns (j), and elements (i,j) of a data frame using [i,], [,j] and [i,j], respectively
5. Extract all observations of a given column (representing a variable) using the $ operator.
6. Summarize the number of observations of a given level of a categorical variable using the table() function
7. Plot basic histograms using hist(), boxplots using boxplot() and scatterplots using plot()

That was a lot of skills covered in the first class. If this is your first time using R, congratulations! That's a great start, and we are going to keep working on those skills. If you're an experienced R user, hopefully it was a good refresher.

This lab has 2 main objectives: **to reinforce and develop new skills**, and to explore some of the **probability** knowledge you have acquired in class.

Please read the whole document, and write the code as you read it. Don't jump directly to the questions, you might miss important info!

Data

Instead of having you enter data, I will be providing the data we are using in this lab on Canvas. Download RootShootRatio_Ma_NEE2021.csv and import this file into R using read.csv() and placing in an object named 'dat'.

```
##   latitude longitude ratio vegetationType
## 1       -3       -60 0.299         forest
## 2       -3       -60 0.411         forest
## 3       -3       -60 0.277         forest
## 4       -3       -60 0.366         forest
```

```
## 5       -3       -60 0.299         forest
## 6       -3       -60 0.411         forest
```

```
dat<-read.csv('~/EEB411_560/L2/RootShootRatio_Ma_NEE2021.csv')
```

Remember! Your code will be different, depending on where your file is located. It's always important to know where a file is located! If you have used R before (or are comfortable with it so far), remember that you can also change your working directory to the folder where the file is located, and simply read the file! But it's OK to do it the way we have been doing it so far!

For more context, the dataset comes from a Nature Ecology and Evolution paper by Ma et al., which aims to investigate environmental drivers of the variation in root-mass fraction across vegetation sampling plots globally. Ma et al

There are 4 columns in this .csv data sheet: * latitude: which represents latitude in decimal degrees * longitude: which represents longitude in decimal degrees * ratio: the root-to-shoot biomass ratio * vegetationType: habitat type (categorical variable; 3 levels: grass, shrub, forest)

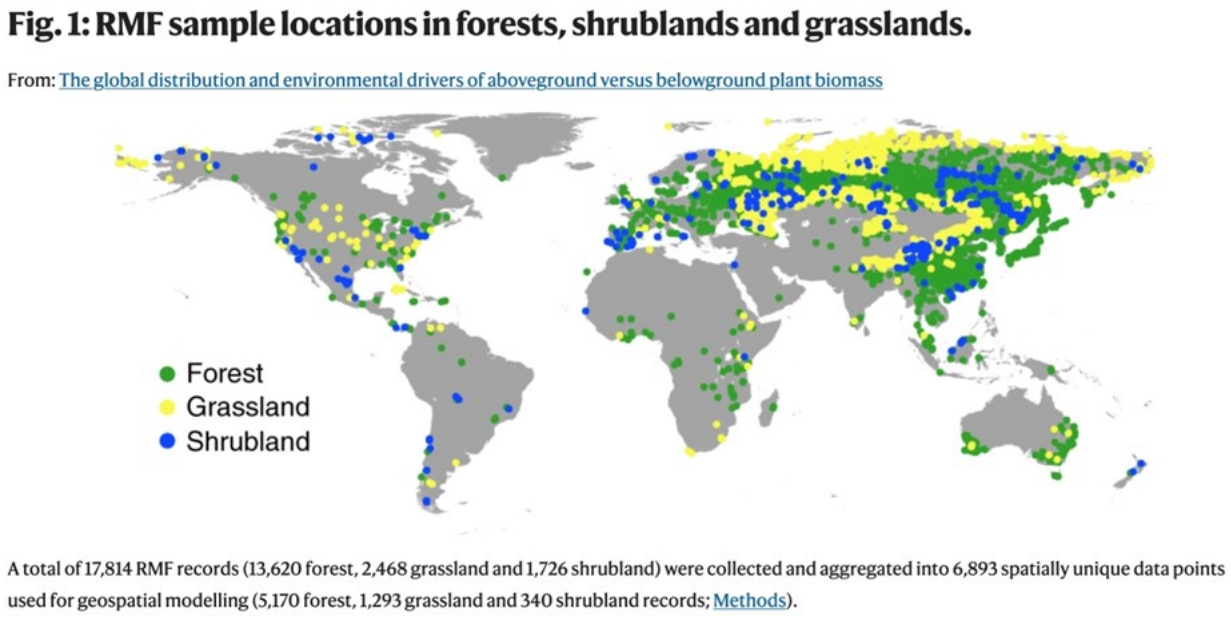


Figure 1: from Ma et al. (2021) Nature Ecology and Evolution showing the distribution of data points or records used to generate data points analyzed in the paper

2.1 Exploring the data

Like last week, use *summary()* to explore the data frame. This is always the first thing I do.

```
summary(dat) # what do you see?
```

```
##     latitude       longitude          ratio          vegetationType
##  Min.   :-51.68   Min.   :-179.43   Min.   :  0.0000   Length:17549
##  1st Qu.: 24.38   1st Qu.:  24.57   1st Qu.:  0.2080   Class :character
##  Median : 38.35   Median :  93.65   Median :  0.2757   Mode  :character
##  Mean   : 31.60   Mean   :  59.65   Mean   :  1.0477
##  3rd Qu.: 52.00   3rd Qu.: 117.04   3rd Qu.:  0.5069
```

```
##  Max.   : 80.63   Max.   : 179.90   Max.   :126.6000
```

Before continuing, look at the summary. Think to yourself, what type of data is each variable? Look at the max and min values for each variables. Do they make sense? You might notice that the minimum value of 'ratio' is 0. Does that make sense? Each observation is the ratio of root to shoot biomass for an entire sampling plot. So a zero value means all plants within that plot have no root biomass? Not sure if I trust that, or if that is possible. We'll get back to that later...

the *summary()* function can be a great way to explore if your data makes sense. If you saw an unusually small or large value for a variable, it could mean an error when transcribing the data! I am sure some of you realized that during the first lab. It's also a great function to run to initially explore the data

Now, remember that we can use the **$** symbol to explore specific variables? Let's do that with vegetation type! The **unique()** function gives you all the unique values of a variable! Try:

```
unique(dat$vegetationType)
```

```
## [1] "forest" "shrub"  "grass"
```

It is good practice to define a categorical variable (vegetationType) explicitly as such:

```
dat$vegetationType <- as.factor(dat$vegetationType)
```

Another thing you can do is to compute the number of observations associated with each of the unique vegetation types (i.e., habitats).

Question 1. 1 pts. Write code to compute the number of observations associated with each vegetation type. Submit the code, plus explicitly write the number of each vegetation type

hint for question 1: Check the first lab!

```
summary(dat)
```

```
##     latitude        longitude          ratio          vegetationType
##  Min.   :-51.68   Min.   :-179.43   Min.   :  0.0000   forest:13499
##  1st Qu.: 24.38   1st Qu.:  24.57   1st Qu.:  0.2080   grass : 2341
##  Median : 38.35   Median :  93.65   Median :  0.2757   shrub : 1709
##  Mean   : 31.60   Mean   :  59.65   Mean   :  1.0477
##  3rd Qu.: 52.00   3rd Qu.: 117.04   3rd Qu.:  0.5069
##  Max.   : 80.63   Max.   : 179.90   Max.   :126.6000
```

There are 13,499 forest, 2,341 grass, and 1709 shrubs.

During Monday's class, I mentioned cross-tabulations. In this, you can summarize the number of observations associated with each level of a categorical variable. However, cross-tabulation summarizes the number of observations associated with pairs of levels across two categorical variables.

Confusing? No worries. As an example, one question we may have on the dataset is whether the relative

frequency of observations across forest, shrubland, and grassland are similar in the tropics vs. temperate zones. Basically we want to count the number of forest, shrubland, and grassland observations in the tropics, and then separately in the temperate zone. This is a cross-tab, because we want to summarize # of observations across two categorical variables, i.e., vegetation type (forest, shrubland and grassland) and climatic zone (tropical vs. temperate).

There is no climate zone variable in our dataset, but we can create it. Tropics is sometimes defined as the region between the Tropic of Cancer **(23.43621°N)** and the Tropic of Capricorn **(23.43621°S)**. We can define a new variable called "zone" and assign observations with latitude < 23.43621 and latitude > -23.43621 as tropical, and those outside this zone, as temperate.

First, we make a new column pertaining to the 'zone' variable. Remember that in order to "create" an object we use **<-**. In this case, we are simply creating a **new column** to our dat dataframe.

```
dat$zone <- NA  # this defines all observations as NA
```

Let's look at this! Let's run:

```
head(dat)
```

```
##   latitude longitude ratio vegetationType zone
## 1       -3       -60 0.299         forest   NA
## 2       -3       -60 0.411         forest   NA
## 3       -3       -60 0.277         forest   NA
## 4       -3       -60 0.366         forest   NA
## 5       -3       -60 0.299         forest   NA
## 6       -3       -60 0.411         forest   NA
```

We see we created a new column, and it has NAs! This is a great skill to have. Oftentimes you will add columns or rows to a dataframe. Other times you will create "empty" objects that you can later populate. NA's are a good way to create those objects.

Now, let's populate this new column! Make sure you understand this section, it will be important later on!

```
dat$zone[dat$latitude<=23.43621 & dat$latitude>=-23.43621] <- 'tropical'
dat$zone[dat$latitude>23.43621 | dat$latitude < -23.43621] <- 'temperate'
head(dat)
```

```
##   latitude longitude ratio vegetationType     zone
## 1       -3       -60 0.299         forest tropical
## 2       -3       -60 0.411         forest tropical
## 3       -3       -60 0.277         forest tropical
## 4       -3       -60 0.366         forest tropical
## 5       -3       -60 0.299         forest tropical
## 6       -3       -60 0.411         forest tropical
```

Some things are happening here: We are populating the column "dat$zone" and it is based on the information on "dat$latitude".

Check the following table, maybe it will help you understand this code! If you have a hard time understanding what the code is doing, approach me or approach Brian or Anchal (either during lab or later). Understanding this will be fundamental later on :)

Operator	Description
<	less than
<=	less than or equal to

Operator	Description
>	more than
>=	more than or equal to
==	equal to
!=	different than
!x	not x
x & y	x AND y
isTRUE(x)	test if X in true
X%in%Y	is X in Y?

Now, we need to define zone as a categorical variable. Again, because we are updating an object, we need to use <-:

```
dat$zone <- as.factor(dat$zone)
summary(dat) # look at the numbers under zone. Compare w/ Fig. 1. Makes sense?
```

```
##     latitude        longitude           ratio          vegetationType
##  Min.   :-51.68   Min.   :-179.43   Min.   :  0.0000   forest:13499
##  1st Qu.: 24.38   1st Qu.:  24.57   1st Qu.:  0.2080   grass : 2341
##  Median : 38.35   Median :  93.65   Median :  0.2757   shrub : 1709
##  Mean   : 31.60   Mean   :  59.65   Mean   :  1.0477
##  3rd Qu.: 52.00   3rd Qu.: 117.04   3rd Qu.:  0.5069
##  Max.   : 80.63   Max.   : 179.90   Max.   :126.6000
##         zone
##  temperate:14690
##  tropical : 2859
##
##
##
##
```

Look at the numbers under zone. Compare w/ Fig. 1. Makes sense? Remember we used latitude to create the zones!

now run the cross-tab using xtabs()

```
xtabs( ~ vegetationType + zone, data=dat)
```

```
##                zone
## vegetationType temperate tropical
##         forest     11425     2074
##         grass       2303       38
##         shrub        962      747
```

to convert frequencies into relative frequencies, we can divide the numbers in each # column by the column sums (which is # of temp vs trop observations)

```
sums<-matrix(c(14690,14690,14690,2859,2859,2859), nrow=3, ncol=2, byrow=F)
sums
```

```
xtabs( ~ vegetationType + zone, data=dat)/sums
```

Run it, but more importantly, try to understand what's happening!

Logical Operators Now that we are done with that code, let's go back to the logical operators. Look at the table with the operators.

Run the following code:

```
vec <- -5:5
vec
```

```
##  [1] -5 -4 -3 -2 -1  0  1  2  3  4  5
```

Ask yourself, how many of those numbers are equal or higher to 4. After that, run the following:

```
vec>=4
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
```

You should have gotten the same answer, right?

We can also ask R to print FALSE or TRUE for each element

```
vec[vec>=4]
```

```
## [1] 4 5
```

Print the True elements:

```
vec[vec>=4]
```

```
## [1] 4 5
```

or tell us which ones are the true elements:

```
which(vec>=4)
```

```
## [1] 10 11
```

*In this case it was the 10th and 11th element of the vector!

If we want to highlight elements >=4 and elements <=-4, we use:

```
vec>=4 | vec<=-4
```

```
##  [1]  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
```

And we can do everything we did previously as well!

2.2 Central tendency and dispersion measures

Now is time to estimate mean, median, standard deviation, and IQR of the sample.

First, let's think how many observations are there in this dataset? Think: what is an observation? is each observation or data point an individual row in the dataset, or is it a column?

Question 2. 1 pts. Write code to compute the the total number of observations in this data frame. Also, write the number of observations

```
nrow(dat)
```

```
## [1] 17549
```

There are 17549 observations of data.

Sometimes, we want to calculate basic summary metrics of central tendency and dispersion.

To calculate the mean root-to-shoot ratio in the dataset, we can type:

```
mean(dat$ratio)
```

```
## [1] 1.047685
```

To calculate the median root-to-shoot ratio in the dataset, we can type:

```
median(dat$ratio)
```

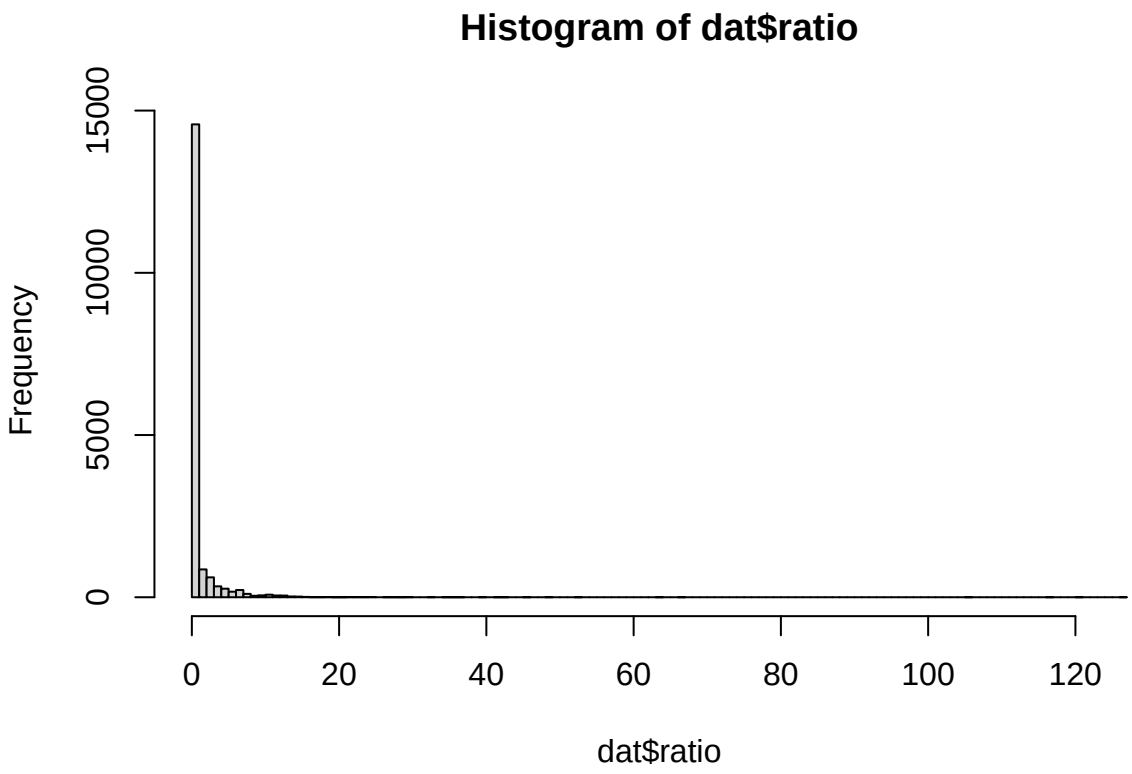
```
## [1] 0.2756757
```

Notice that the mean is larger than the median.

Question 3. 2 pts. *Part 1*: What does the mean and median values suggest about the skewness of the distribution of root-to-shoot biomass ratio? Explain why (I uploaded the lecture slides to canvas. They might be a good resource for this question!). *Part 2* Write code to generate a histogram of the distribution of root-to-shoot biomass ratio values? Does the histogram support your answer in Part 1? Why? Try a different number of breaks (maybe high numbers?). No need to upload the plot, just the code

Part 1. Since the median is much smaller than the mean, this suggests that the data is signficantly skewed right. Part 2. The histogram below confirms that the data is skewed right and a lot of the ratios are numbers close to 0, but some ration are above 100.

```
hist(dat$ratio,breaks =120)
```



Besides the mean and median, you can calculate measures of dispersion such as sample standard deviation. To do that, you can use the function sd():

```
sd(dat$ratio)
```

```
## [1] 3.108265
```

You can also calculate the interquartile range (IQR; the difference between 75th and 25th percentile values):

Question 4. 1 pts. Write code to compute the IQR of root-to-shoot biomass ratio (hint: the function to compute percentile values is quantile(). And the IQR is the Q3 - Q1

```
Q = quantile(dat$ratio)
IQR = Q[4] - Q[2]
as.numeric(IQR)
```

```
## [1] 0.2989441
```

```
Q
```

```
##          0%         25%         50%         75%        100%
##   0.0000000   0.2079707   0.2756757   0.5069148 126.6000000
```

The IQR of the root to shoot biomass ratio is 0.2989441.

Struggling with Q4? Remember to check lecture slides, the html I shared for lecture 2 and the shared code. You can also check ?quantile to see the R help file

2.3 Summarizing values of a continuous variable across different levels of a categorical variable

We used table() to summarize frequencies for one categorical variables and we used cross-tabulation via R function xtabs() to summarize frequencies across two categorical variables.

What if we want to summarize values of a continuous variable across different levels of a categorical variable? For example, we want the mean root-to-shoot biomass ratio (i.e., the continuous variable) for each level of vegetationType (i.e., the categorical variable). So we would want to calculate the mean root-to-shoot ratio value for each of the three levels: (a) forest, (b) grass; (c) shrub.

We can use the aggregate() function to do this:

```
aggregate(ratio ~ vegetationType, FUN='mean', data=dat)
```

```
##   vegetationType     ratio
## 1         forest 0.3357208
## 2          grass 5.1181975
## 3          shrub 1.0955096
```

where we get mean values for the ratio for forest, gradd, and shrub!

Question 5. 2 pts.Write code to compute the standard deviation of root-to-shoot ratios for forest, grass, and shrub using the aggregate() function

See code below.

```
aggregate(ratio ~ vegetationType, FUN='sd', data=dat)
```

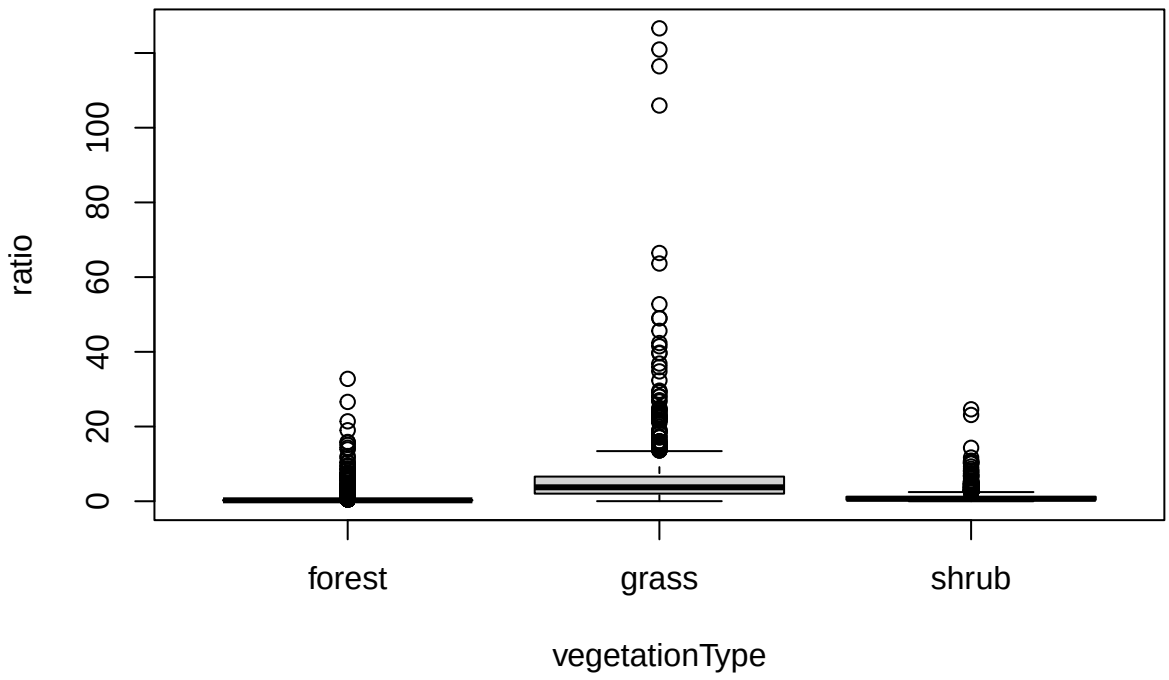
```
##   vegetationType    ratio
## 1         forest 0.673928
## 2          grass 6.972619
## 3          shrub 1.531630
```

2.4 Visualizing values of a continuous variable across different levels of a categorical variable

In Monday's lecture, we (very briefly) discussed the advantages of using a boxplot to visualize values of a continuous variable across different levels of a categorical variables over a barplot. Simply put, boxplots provide more graphical information about the shape of the distribution then barplot. Therefore it should be the first plot we consider.

Let's plot one:

```
boxplot(ratio ~ vegetationType, data=dat)
```



For the boxplot: notice that the syntax is very similar to aggregate(). Does the syntax make sense? Read it aloud: (box)plot ratio as a function of vegetationType ‘ ’ equivalent to ‘as a function of’

Question 6. 1 pts. Does the distribution of root:shoot ratio for each habitat look symmetric? How would you describe the skew of the distributions?

Each distribution is not symmetric. The ratio for each vegetation is skewed right.

For ratios, the classic transformation would be to log-transform it. The base of the log doesn't matter, so we can use the default log (natural). Basically log-transform for ratios has a nice property of being symmetrical. Later on, we will learn more about transforming data!

Try the following code.

```
log(1/2)
log(2/1)
```

What do you notice about them? That might give you a hint as to why we transform ratios using log.

Now let's generate a new boxplot to explore our transformed values (and to answer Q 7!)

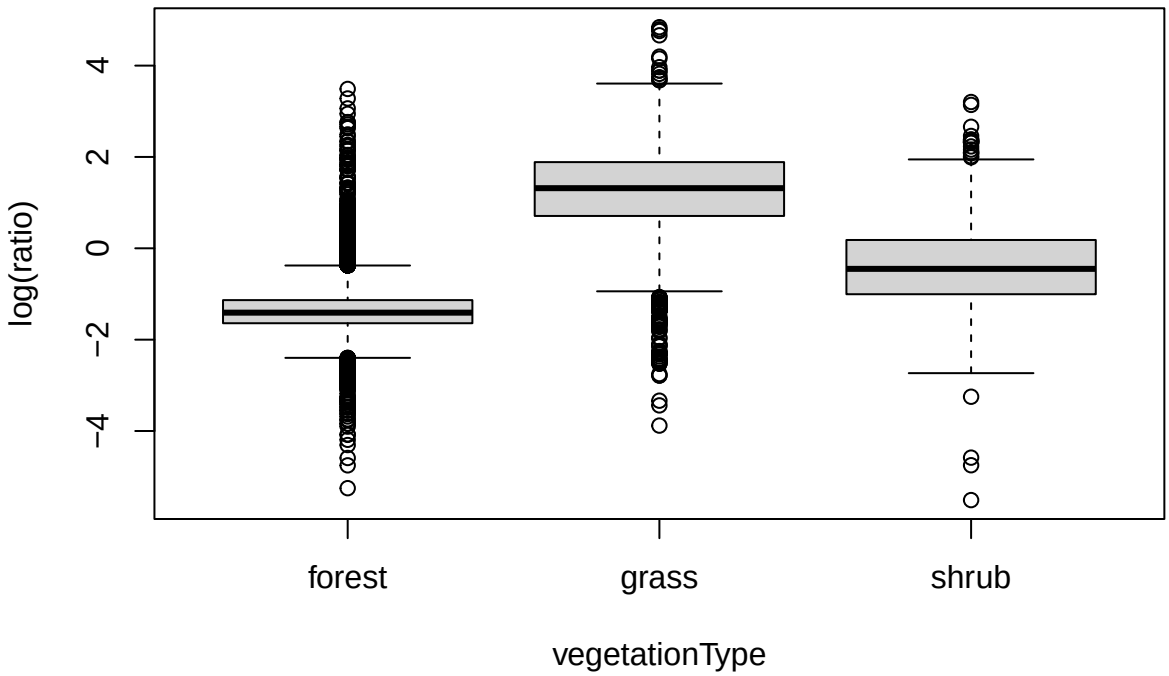
Question 7. 1 pts. Does the distribution of log(root:shoot ratio) for each habitat look more symmetric than the distribution of untransformed root:shoot ratio for each habitat?

Based on the below graph, the transformed data boxplot appears more symmetric than the untransformed data's boxplot

The code to run the log transformed boxplot is the following:

```
boxplot(log(ratio) ~ vegetationType, data=dat)
```

```
## Warning in bplt(at[i], wid = width[i], stats = z$stats[, i], out =
## z$out[z$group == : Outlier (-Inf) in boxplot 3 is not drawn
```



Are you getting an error message? Keep reading!

But! you're probably getting an error message. Why do you think it is? Try:

```
summary(dat)
```

Any ideas?

How about:

```
min(dat$ratio)
```

Remember? We mentioned those zeroes before! A zero would mean no roots! It is a weird result. What happens when you get try to run the following?

```
log(0)
```

So, let's create a new dataframe with no zeroes.

```
dat_noZeroes <- dat[!dat$ratio %in% 0,]
head(dat_noZeroes)
```

```
##   latitude longitude ratio vegetationType     zone
## 1       -3       -60 0.299         forest tropical
## 2       -3       -60 0.411         forest tropical
## 3       -3       -60 0.277         forest tropical
## 4       -3       -60 0.366         forest tropical
## 5       -3       -60 0.299         forest tropical
## 6       -3       -60 0.411         forest tropical
```

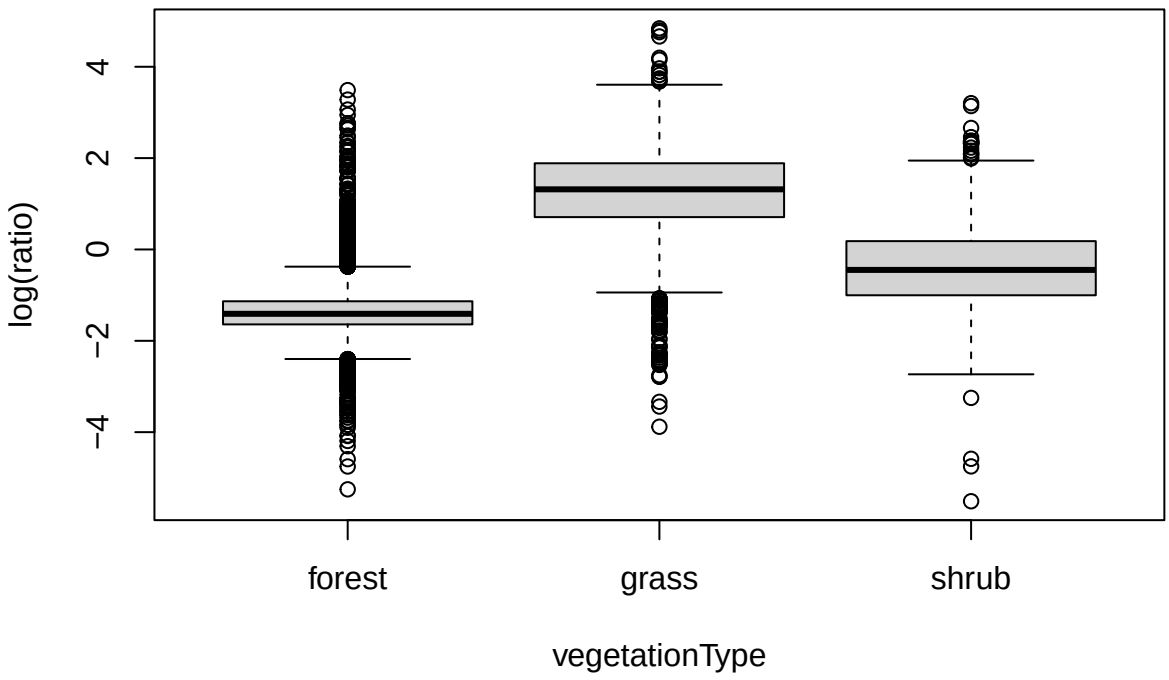
Confused at that code? Check the table with the logical operators! Section 2.1!

Now you can run the boxplot for question 7 (remember, the data should be the NEW dataframe!)

Question 8. 2 pts. Upload a .jpg, .png or .pdf file of the boxplot of log(root:shoot ratio) across different habitats (grass, forest, shrub).

Image of boxplot below and attached

```
boxplot(log(ratio) ~ vegetationType, data=dat_noZeroes)
```



3 Probabilities

To finish this lab, let's do a couple fun exercises.

First, let's simulate one coin toss:

```
rbinom(1,1,0.5)
```

```
## [1] 0
```

Some of you might get a 1 (success), some of you a 0. Let's assume a success is a head (or anything you want, really).

To throw one coin, 1,000 times:

```
rbinom(1000,1,0.5)
```

And to throw a single coin 1,000 times:

```
rbinom(1,1000,0.5)
```

Where the result, is the number of successes.

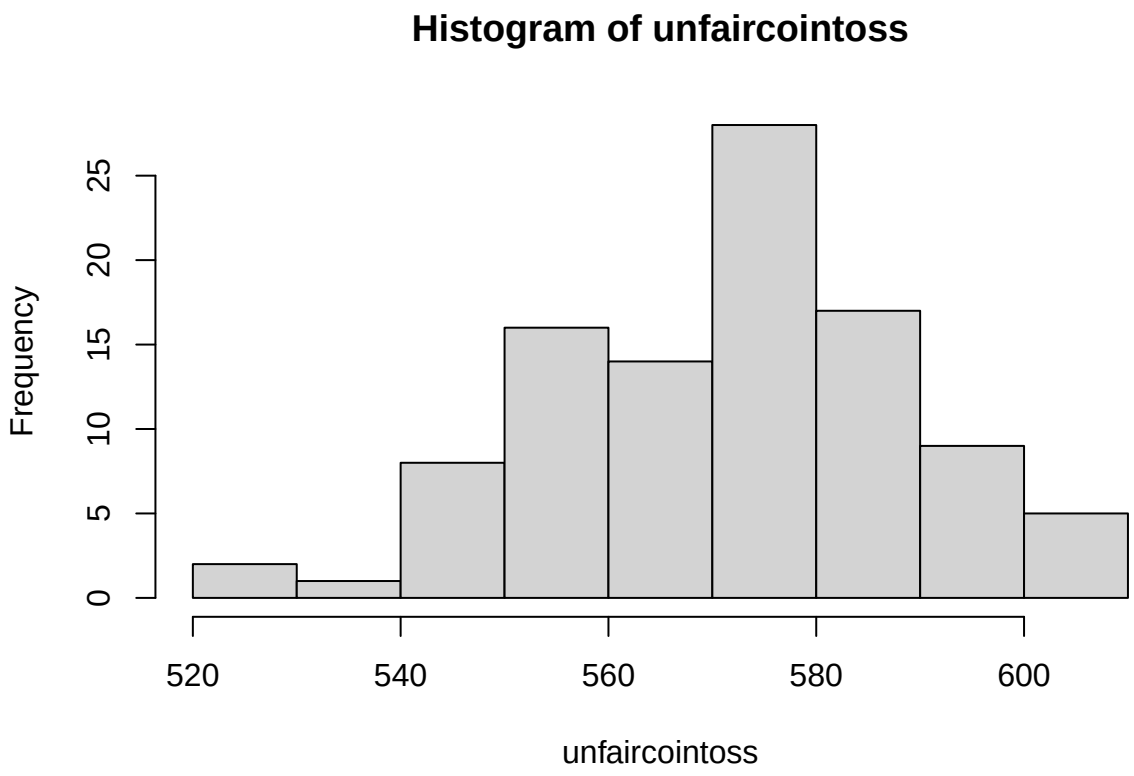
Let's use one example, and create an object with the results from 1,000 experiments, where we tossed a coin 1000 times. Each experiment = 1,000 tosses

```
cointoss<-rbinom(1000,1000,0.5)
```

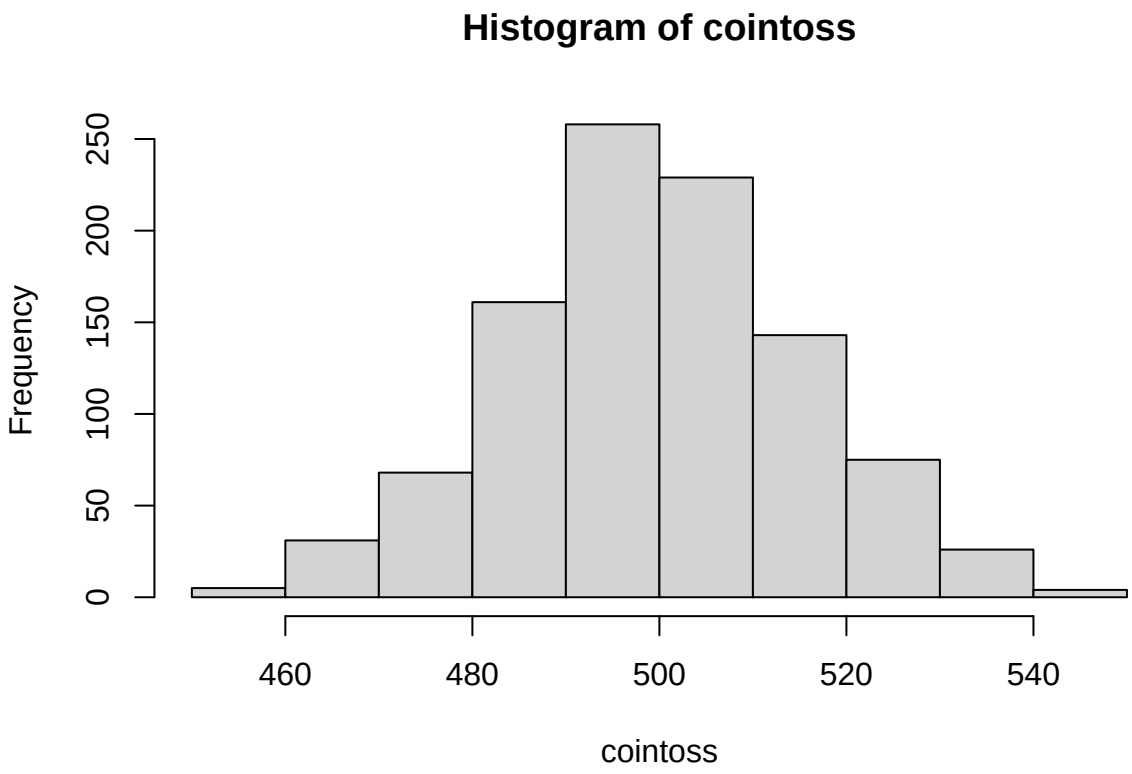
You can a vector, where each element is the number of successes for a particular experiment (1,000 throws)

Question 9. 3 pts. Create another object, in which you run an experiment 100 times, by tossing an **unfair** coin 1,000 times. This coin has a probability of success of 0.57. Name the object "unfaircointoss" Plot both histograms and compare them

```
unfaircointoss = rbinom(100,1000,0.57)
h1=hist(unfaircointoss, breaks=10)
```




```
h2=hist(cointoss, breaks=10)
```



The histogram from the fair coin toss is symmetric, and appears to be centered approxiamately at 500 successes for 1000 coin tosses. The histogram from the unfair coin toss appears less symmetric, likely due to fewer experiment trials. It appears centered at approxiamately 570 successes for 1000 coin tosses.

We can use the command sample() to obtain a random sample from an object. Using the following line, we sample 3 random elements from the cointoss experiment.

```
sample(cointoss,3,replace=T)
```

Question 10. 1 pts Sample the unfaircointoss 4 times. Report your sampled values. Based on the results from that random sample, would you be able to conclude that the coin isn't fair? Why?

```
sample(unfaircointoss,4,replace=T)
```

```
## [1] 589 576 574 589
```

These data points are all larger than the expected mean of an experiement for a fair coin toss. This points to the conclusion that the coin is unfair, but with only a small number of data points, we are not able to be confident in this conclusion.

Because we are simulating random data, your answer to question 10 will look different than someone elses!

To finish this lab, I am going to present you ways in which you can simulate data from other distributions (not just binomial). There are no more questions for this lab, but this might come in handy in the future:

Uniform distribution (all values have same probability of being sampled or simulated). 50 values between 0 and 1:

```
runif(50,0,1)
```

Normal distribution (higher probability near the mean). The spread depends on the standard deviation value. 50 values, when mean is 10 and sd is 2.5:

```
rnorm(50,mean=10,sd=2.5)
```

As you can tell, we can simulate data from very different distributions: rpois (poisson), rweibull(weibull), rmultinom(multinomial), etc etc. This is a great tool! and something that would take a long time to do in excel (or by hand!)