Graphs and distributions

Kevin Healy

2023-09-04

In this session we will focus on plotting. Plotting is a vital step in any analysis, both for the initial stages so you can visually check for issues in your data but also to demonstrate and communicate your results.

Uploading data

We will continue using the lifespan data from week 2 so lets first upload our data.

```
lifespan_data <- read.csv("lifespan_data_28_8_2023.csv",
                          header = T,
                          sep = ",")
```

We will also upload some packages at the start of our session. Here we will download the 'hdrcde' package which will allow us to calculate the mode, the 'vioplot' package for violin plots and 'moments' package will allow us to calculate skewness.

```
#Install using this line without the # at the start
# if you have not already install this package
#install.packages("hdrcde")

#open up the package
library(hdrcde)
```

```
## This is hdrcde 3.4
```

```
# install.packages("vioplot")
library("vioplot")
```

```
## Loading required package: sm

## Package 'sm', version 2.2-5.7: type help(sm) for summary information

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
# install.packages("moments")
library("moments")
```

Summary statistics

Summary statistics are when you describe your data before conducting the analysis. Its a good way to communicate what your data looks like using things like the mean, mode, median, range, standard deviation and other metrics.

Lets calulate each of central tendancy metrics for maximum lifespan first.

```
#mean
mean(lifespan_data$maximum_lifespan_yr)
```

```
## [1] 19.51089
```

```
#median (middle value)
median(lifespan_data$maximum_lifespan_yr)
```

```
## [1] 17.4
```

```
#mode
#We will use the function hdrcde
hdr(lifespan_data$maximum_lifespan_yr)$mode
```

```
## [1] 13.32601
```

Notice the slight difference between each of these central tendency values.

Now lets look at measure of variance, including standard deviation, variance and also the range, we can be a very useful way to describe how spread out the data is.

```
#mean
sd(lifespan_data$maximum_lifespan_yr)
```

```
## [1] 12.16588
```

```
#median (middle value)
var(lifespan_data$maximum_lifespan_yr)
```

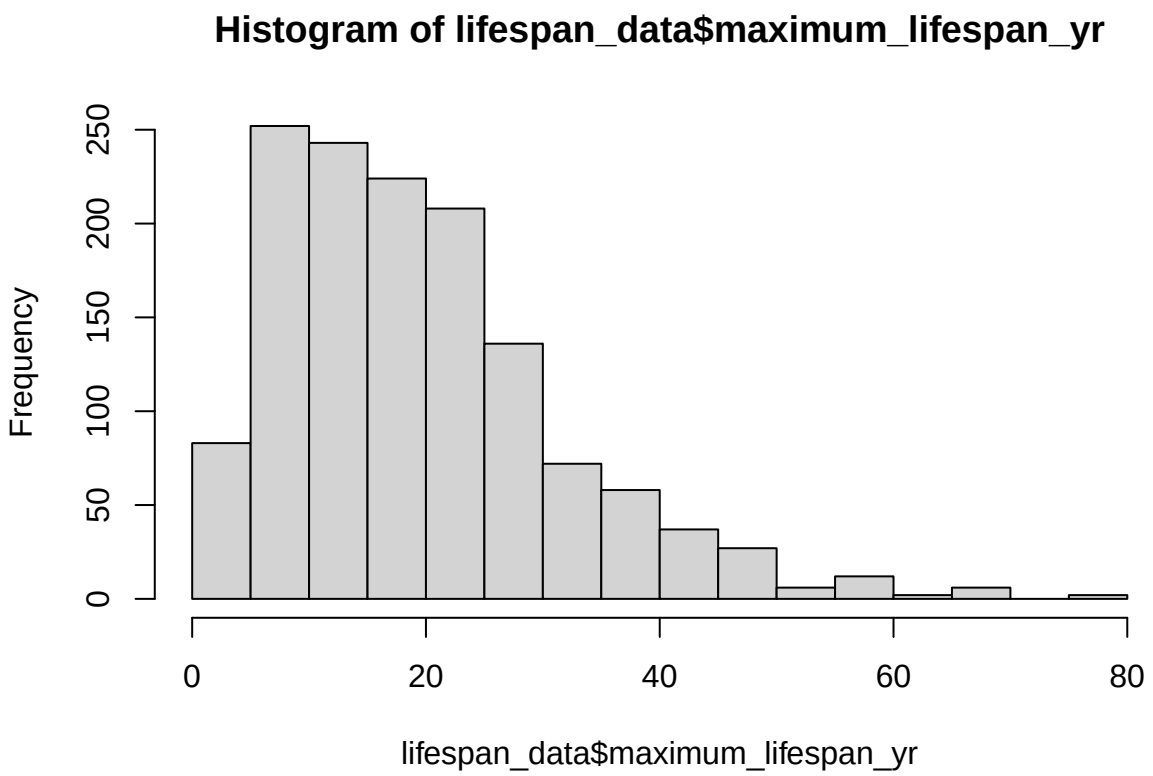
```
## [1] 148.0086
```

```
#range
#the first value gives the lowest value and second value gives the
#highest value
range(lifespan_data$maximum_lifespan_yr)
```

```
## [1]  2.1 79.0
```

Lets now look at the distribution of the data using a histogram

```r
#create a histogram for lifespan data
hist(lifespan_data$maximum_lifespan_yr)
```



Here we can see the how frequent lifespans are. You may notice that the values are not completely normally distributed but instead are postively skewed. For example, we can see that lifespans over 60 yrs are rare while lifespans between 5 and 20yrs are the much more frequent observations and that the spread of data is not symmetrical. While we typically don't report skewness we can calulate it as below.

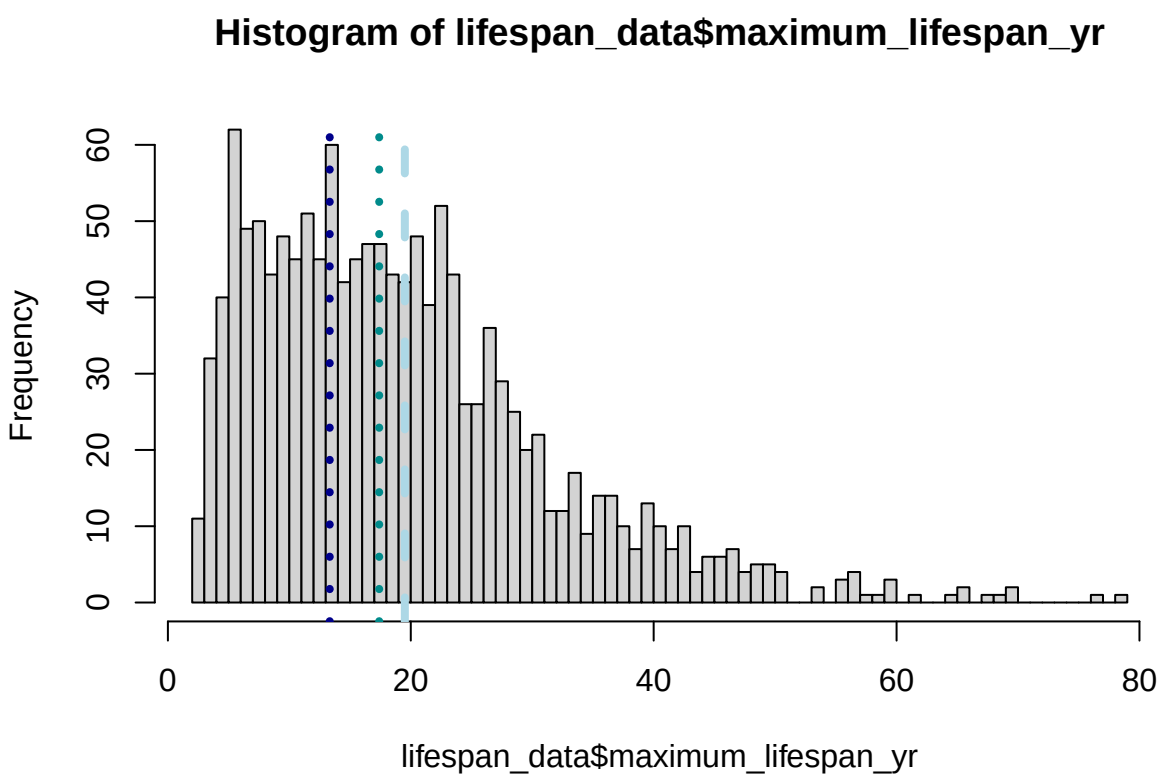
```r
skewness(lifespan_data$maximum_lifespan_yr)
```

```
## [1] 1.183093
```

We can also change the number of bins or breaks for the data if we want more detail and we can also plot the mean, median and mode on the histogram

```r
# We can ask the function to break up the data into
# 100 bins using breaks =
hist(lifespan_data$maximum_lifespan_yr,
     breaks = 100)

#Add a line for the mean
abline(v= mean(lifespan_data$maximum_lifespan_yr),
       lwd = 4,
       lty = 2,
       col = "lightblue")

#Add a line for the mean
abline(v= median(lifespan_data$maximum_lifespan_yr),
```

```
       lwd = 4,
       lty = 3,
       col = "darkcyan")

#Add a line for the mean
abline(v= hdr(lifespan_data$maximum_lifespan_yr)$mode,
       lwd = 4,
       lty = 3,
       col = "darkblue")
```



The non-normal distribution explains why the mean (light blue), median (cyan) and mode (dark blue) are different values.

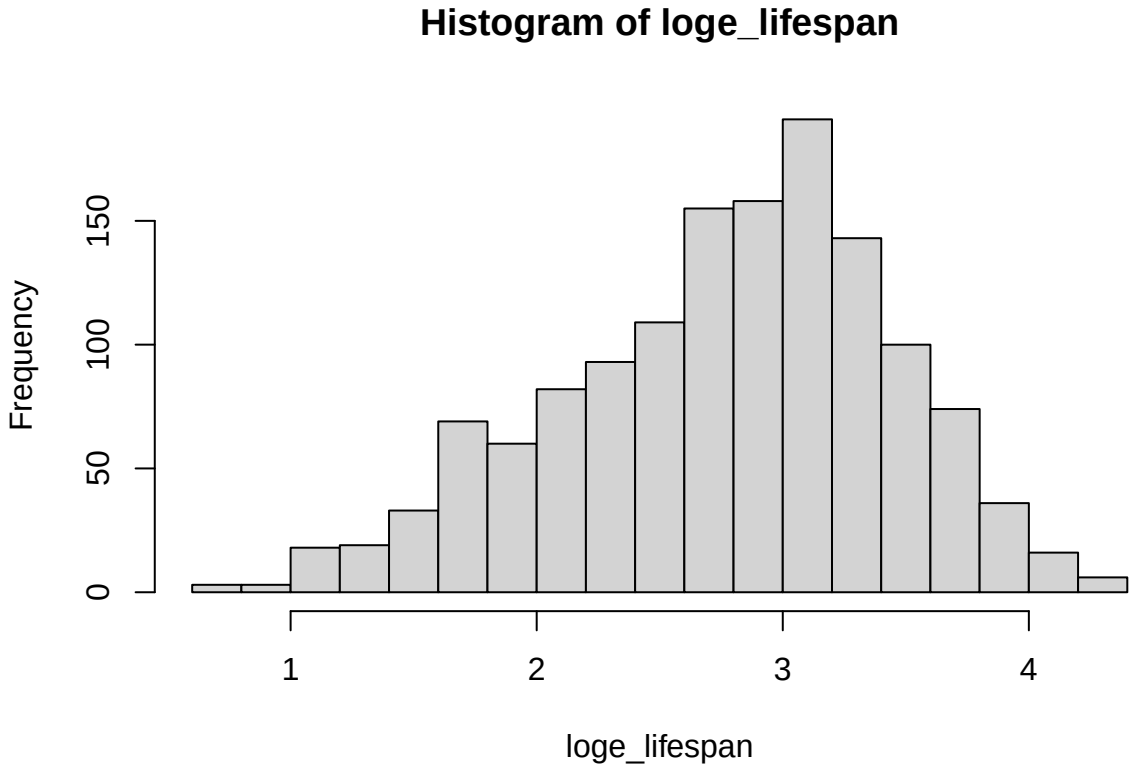
Overall, this data looks like its log-normal which is a type of distribution that appears in biological data a lot.

If we wanted a transformation to make the data look more normal we could use the log() function

```
#we can log to the natural base e
loge_lifespan <- log(lifespan_data$maximum_lifespan_yr)

#or to the base of 10
log10_lifespan <- log10(lifespan_data$maximum_lifespan_yr)
```

If you check the mean, median and mode you will notice they are now more similar and when we do a histogram it will look more normally distributed.

```
hist(loge_lifespan,
     breaks = 20)
```



Notice the x-axis scale is now on a log scale. If we use log10 instead, the scale will be still logged but each unit will indicate an order of magnitude difference in size. That is 10years = 1, 100years = 2, 1000years = 3 etc.

We will see in later session how logs can be a very useful transformation for biological data. For now lets make some figures.

Figures

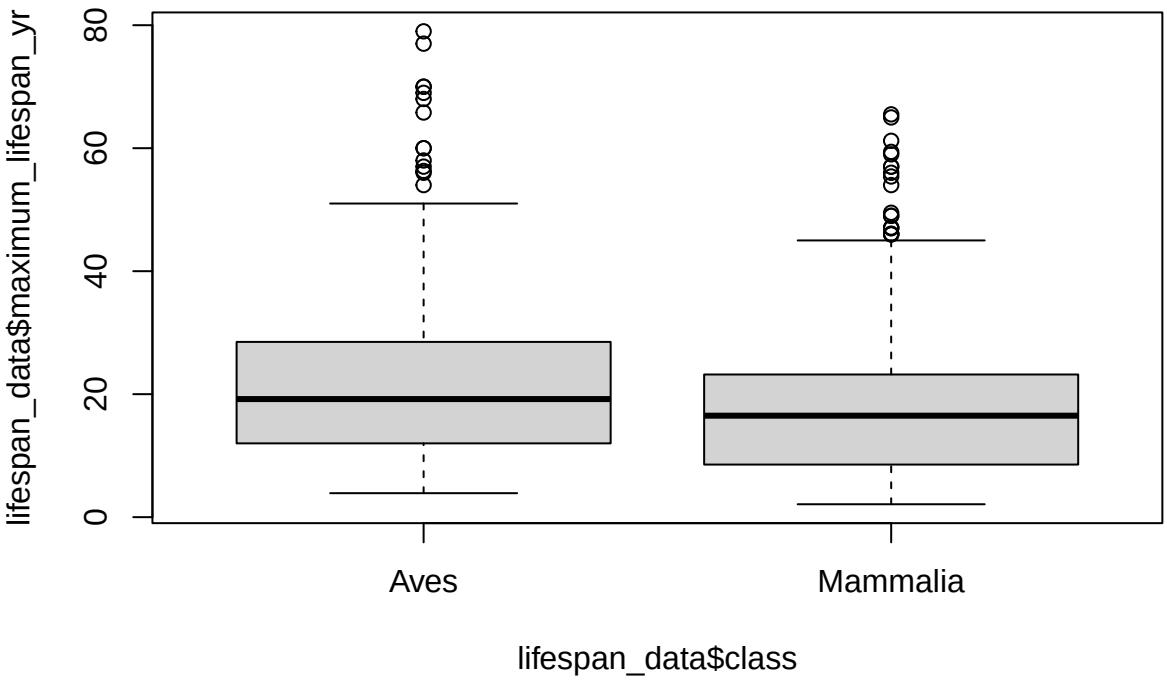
If you remember how to subset data plotting can be straight forward in R. First lets make a boxplot of how lifespan is different between mammals and birds. In all of the plotting we will use the y~x format, were x is the explanitory varable and y is the response variable. This means we think changing the variable on the x-axis will result in a change of the variable on the y-axis. For example, we might think that higher temperatures cause higher ice-cream sales. We don't think higher ice-cream sales cause higher temperature. Hence we would always write it as ice-cream sales ~ temperatures. This will be especially important when we start modelling.

Boxplot

Lets start off with a simply box plot using the Y~ X notation.

Hint when plotting type dev.new() first to create a new window on your desktop. This can make it easier to see what you are doing if your screen is small.

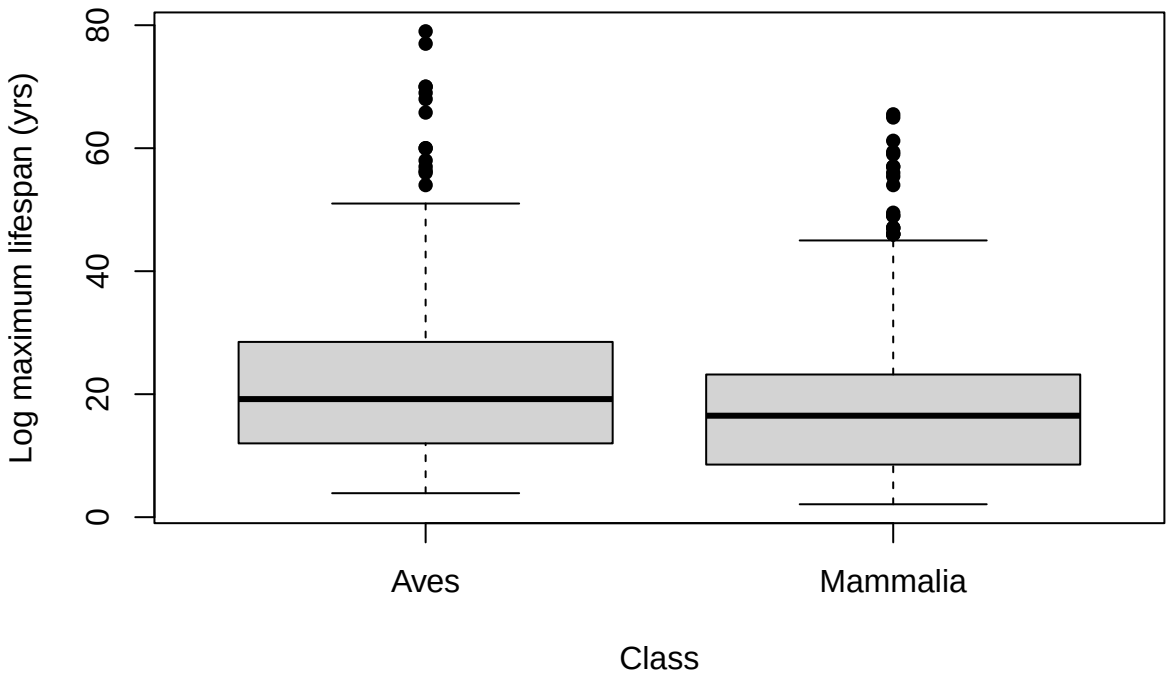
```
boxplot(lifespan_data$maximum_lifespan_yr ~ lifespan_data$class)
```



In the boxplot the middle line is the median, the boxes themselves extend to the upper and lower 25% quartiles, which means if you dived the data into 4 the middle two sections of the data are in those boxes. The long bars extend to the minimum and maximum excluding outliers. Outliers, given as little data points are when a value is greater than the interquartile + 1.5 of the interquartile range (the range of the middle box).

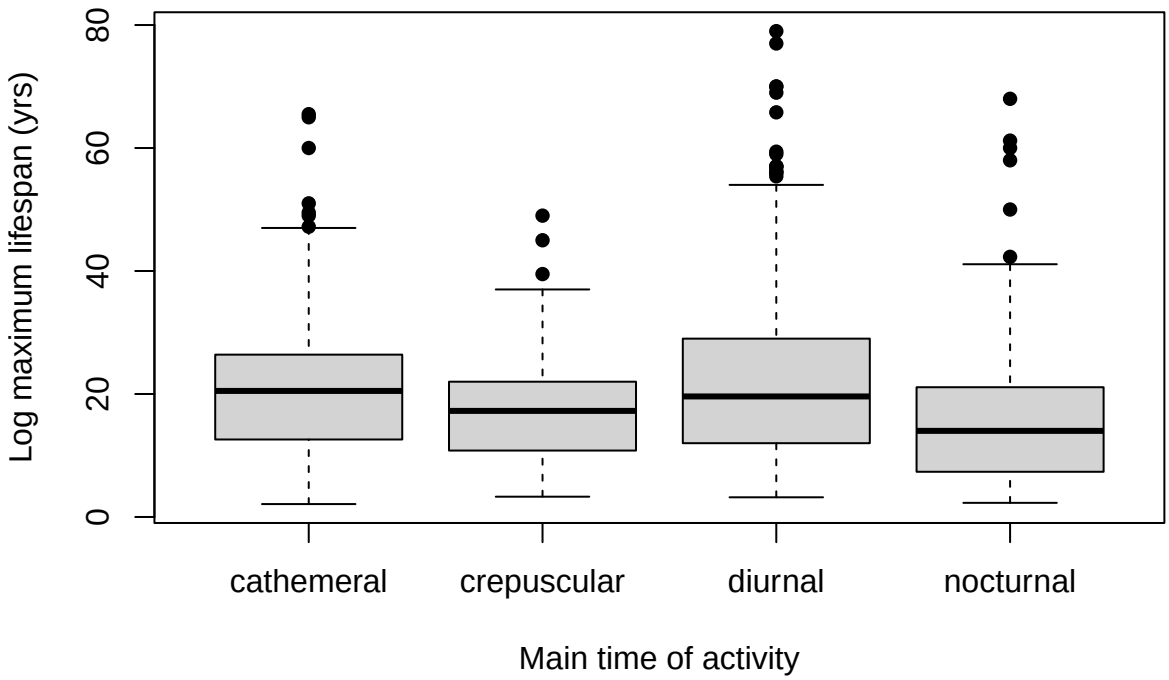
We can use a range of options within the function of both boxplot() and plot() to change data point styles, size etc. Here we use xlab = " " and ylab = " " to put titles on both the axis. We also use pch = 16 which sets the type of point on the graph. If in doubt with these options use ?boxplot

```
boxplot(lifespan_data$maximum_lifespan_yr ~ lifespan_data$class,
  xlab = "Class",
     ylab = "Log maximum lifespan (yrs)",
     pch = 16)
```



Box plots can be used for a more than one group, for example, lets plot the activity time

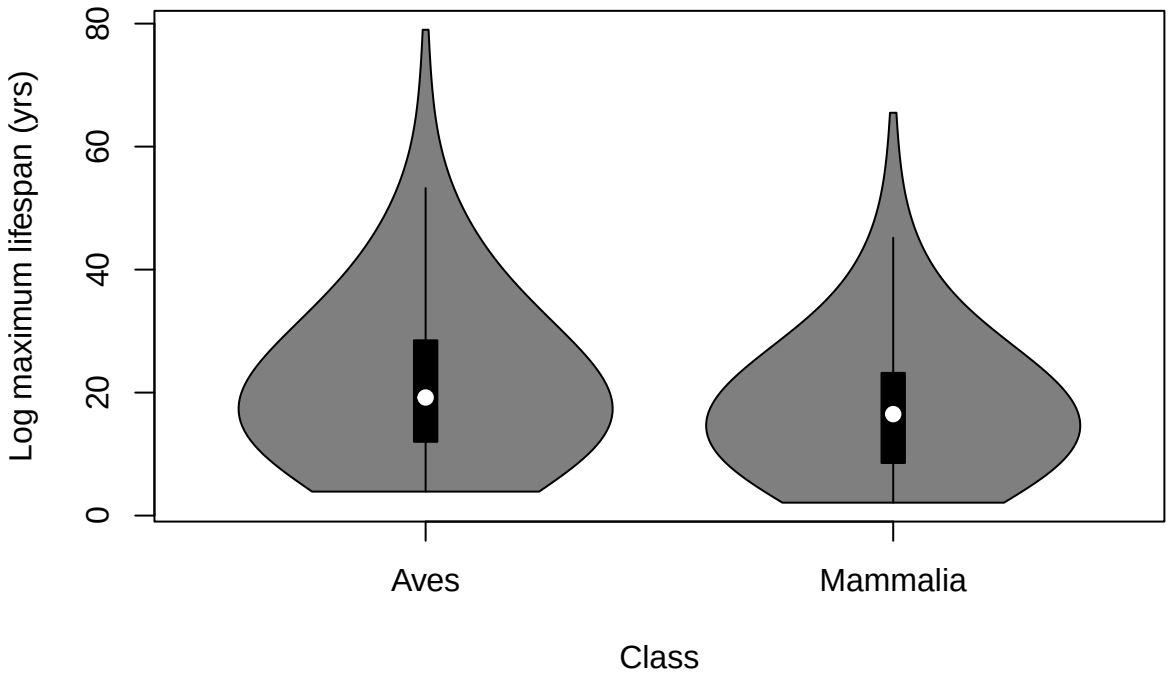
```
boxplot(lifespan_data$maximum_lifespan_yr ~ lifespan_data$daily_activity,
  xlab = "Main time of activity",
    ylab = "Log maximum lifespan (yrs)",
    pch = 16)
```



Violin plots

Violin plots can be thought of as histograms on their side. They can give a better idea of how data is distributed within each of the groups of a boxplot.

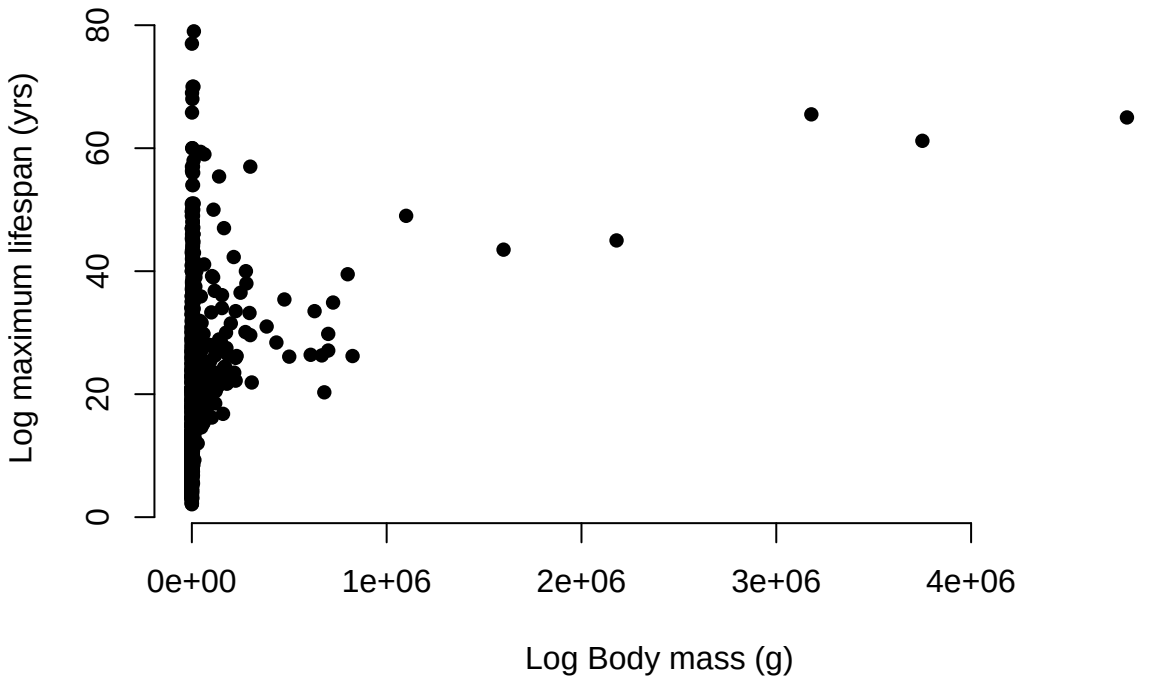
```
vioplot(lifespan_data$maximum_lifespan_yr ~ lifespan_data$class,
     xlab = "Class",
     ylab = "Log maximum lifespan (yrs)",
     pch = 16)
```



Scatterplot

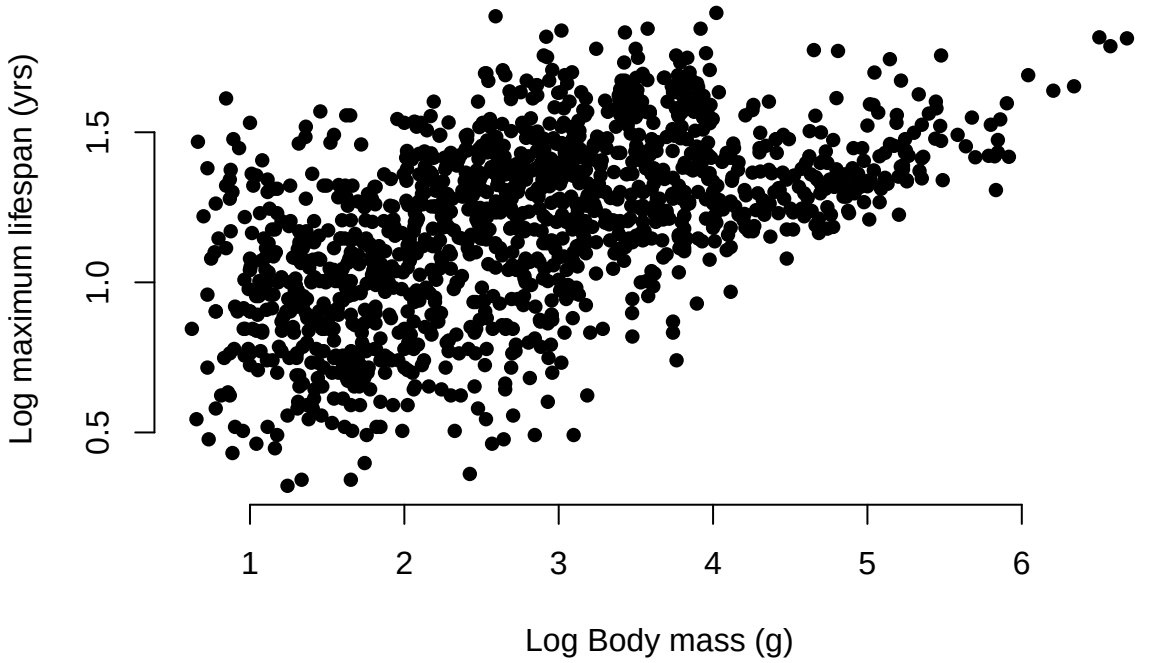
For continuous data we typically use scatter plots often to demonstrate trends of how increasing/decreasing values on the x-axis leads to a increase/decrease on the y-axis. Lets do a scatter plot for lifespan and body mass. Here, we think being larger leads to longer lifespans (due to increased protection) so we put mass on the x axis and lifespan on the y-axis.

```
plot(lifespan_data$maximum_lifespan_yr ~ lifespan_data$mass_g,
     xlab = "Log Body mass (g)",
     ylab = "Log maximum lifespan (yrs)",
     pch = 16,
     bty = "n")
```



Looking at this plot it doesn't look right as all the data points are squashed up in one corner. This is a classic example of the issues of using log-normal data as comparing a 1 gram mouse to a 10 ton elephant can cause issues with trying to fit them on the same graph. One way to deal with this is a log transformation.

```
plot(log10(lifespan_data$maximum_lifespan_yr) ~ log10(lifespan_data$mass_g),
     xlab = "Log Body mass (g)",
     ylab = "Log maximum lifespan (yrs)",
     pch = 16,
     bty = "n")
```



We might also want to highlight different groups in our data. One quick way to do this is plot over the graph

with just the data points but now with different colours using the points() function. This just plots points following the same y~x structure in the plot() function.

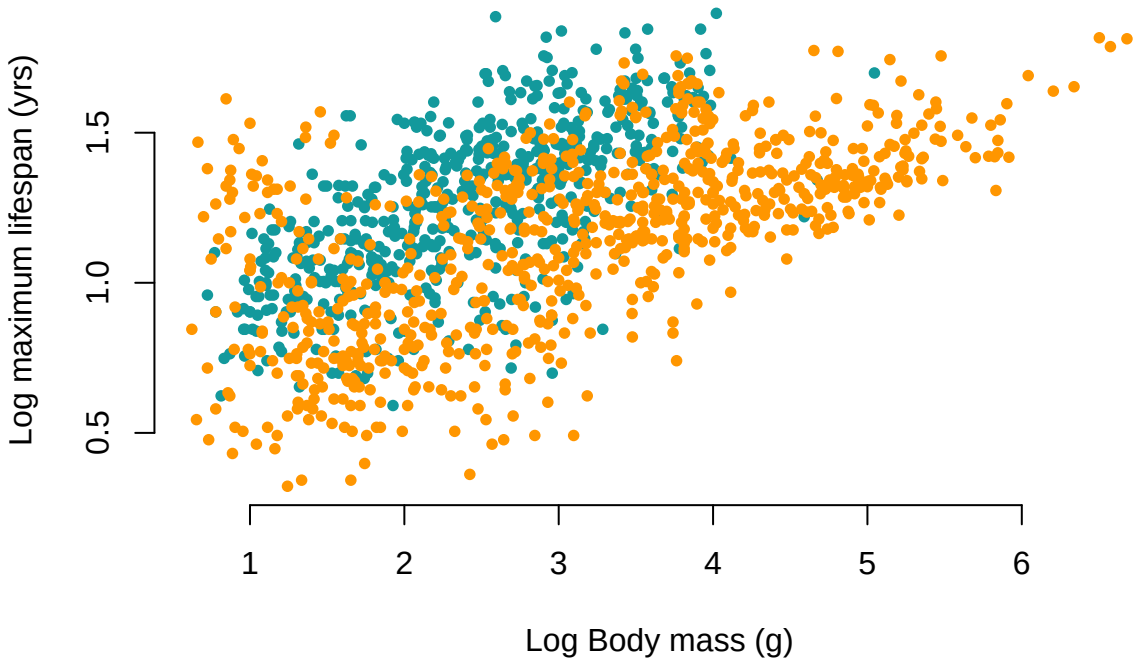
We can also use more arguments for our plotting. The cex argument will change the size of the points, cex = 0.5 will half their size, cex = 2 will double the point size and so on.

We can also create colours using default colours such as "red", "blue", "cyan" etc that are already loaded into R (See here for colours by name https://bookdown.org/hneth/ds4psy/D-3-apx-colors-basics.html). If you have the HEX number for a colour you can do the same thing with a # in front of it and in quotation marks such as "#13999C".

You can also use the rgb() function to specify a colour based on its red green blue value. You can find rgb numbers and HEX values from online palettes such as colourlovers (https://www.colourlovers.com/palette).

There are a number of other ways and palettes to use for colours in R so don't be afraid to google and try things out.

```
#notice how I pick the colour using the word here
plot(log10(lifespan_data$maximum_lifespan_yr) ~ log10(lifespan_data$mass_g),
     xlab = "Log Body mass (g)",
     ylab = "Log maximum lifespan (yrs)",
     pch = 16,
     bty = "n",
     cex = 0.8,
     col = "white")

#We can create a list of all Aves maximum lifespan
aves_life <- lifespan_data[lifespan_data$class == "Aves", "maximum_lifespan_yr"]

#We can create a list of all Aves body mass
aves_mass <- lifespan_data[lifespan_data$class == "Aves", "mass_g"]

#In this case we use
points(log10(aves_life) ~ log10(aves_mass),
       pch = 16,
       col = "#13999C",
       cex = 0.8)

#Now lets do it for mammals

#We can create a list of all Aves maximum lifespan
mam_life <- lifespan_data[lifespan_data$class == "Mammalia", "maximum_lifespan_yr"]

#We can create a list of all Aves body mass
mam_mass <- lifespan_data[lifespan_data$class == "Mammalia", "mass_g"]

#We will use the rgb() function to specify an orange colour
points(log10(mam_life) ~ log10(mam_mass),
       pch = 16,
       col =  rgb(255,150,0, maxColorValue = 255),
       cex = 0.8)
```

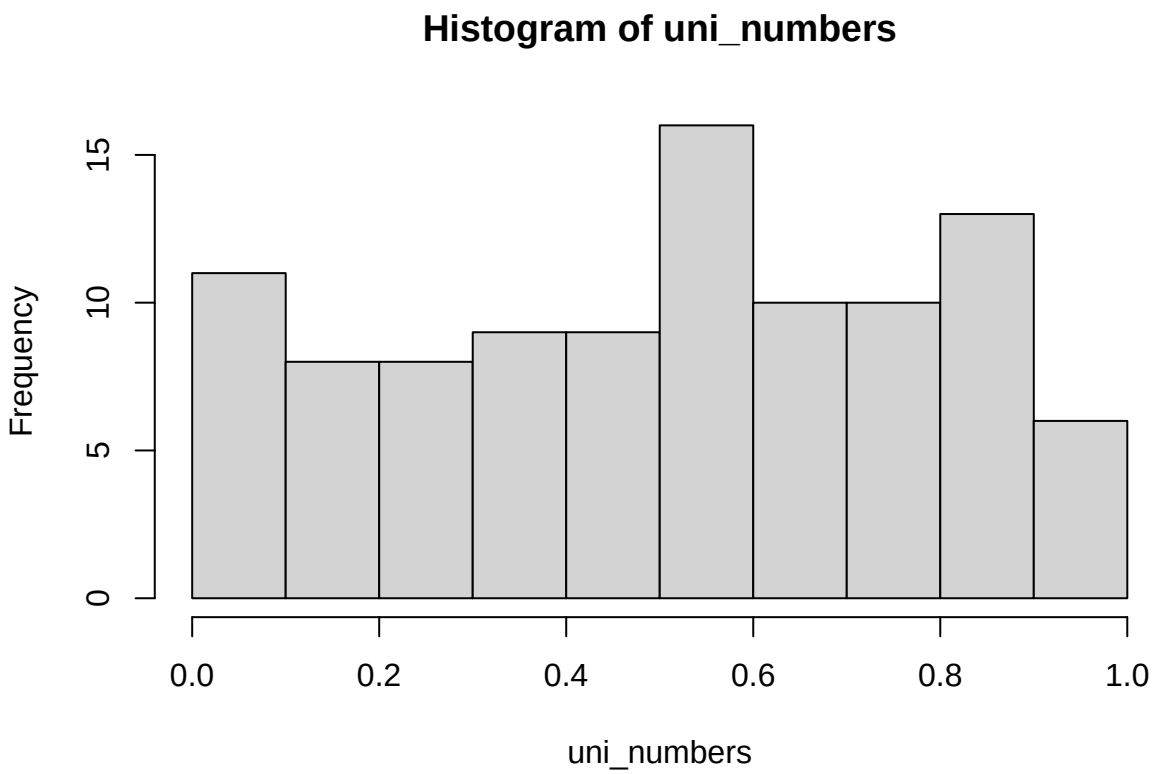


Other types of distributions

One of the things we can do in R is simulate distributions by using functions that will generate a number of values according to some input paremeters for the given distribution.

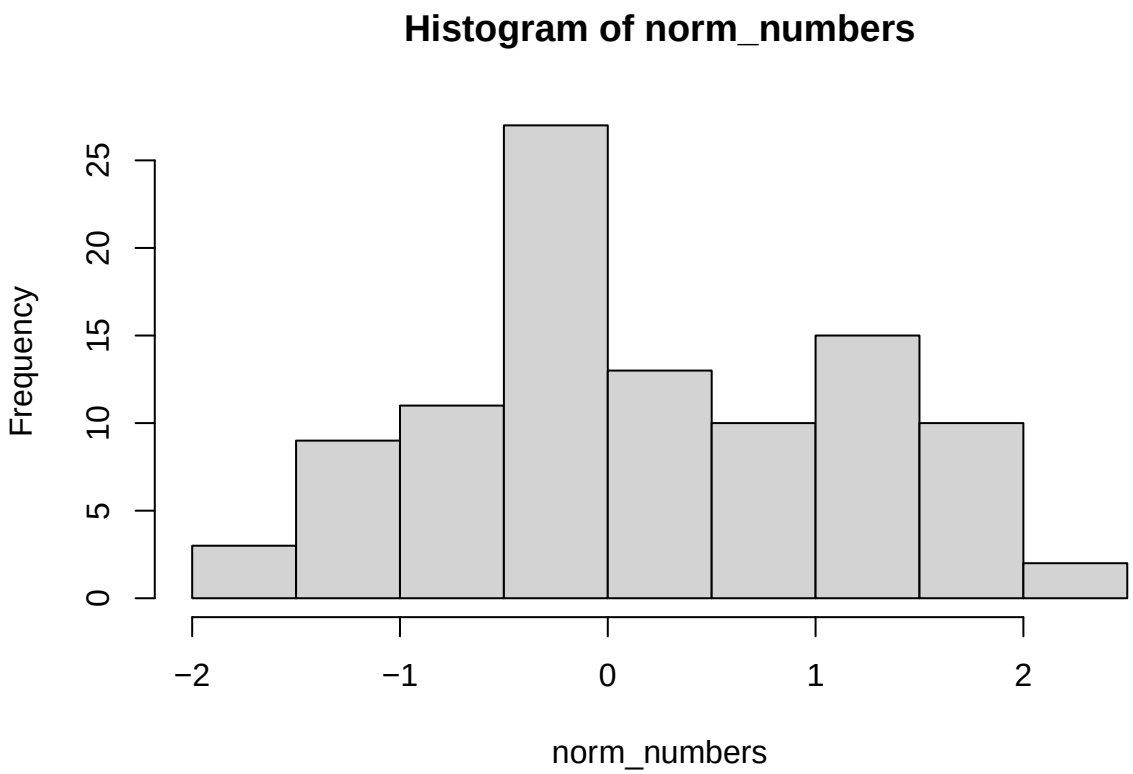
For example, we can create a uniform distribution be simulating 100 random numbers between 1 and 10 using runif()

```
#generate values for a uniform distribution
uni_numbers <- runif(n = 100, min = 0, max = 1)

#lets plot a histogram of that
hist(uni_numbers)
```



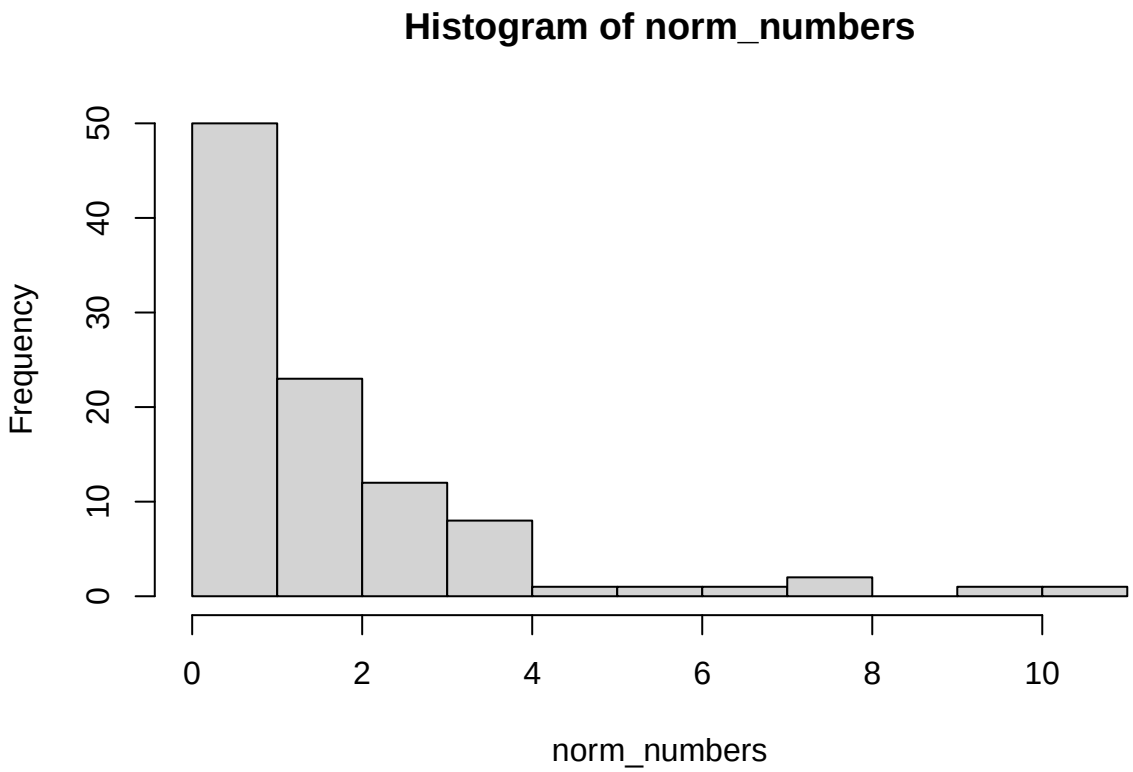
We can also plot a normal distribution using rnorm(), all we need to tell it is how many values we want to simulate, the mean and the standard deviation of the distribution.

```
#generate values for a normal distribution
norm_numbers <- rnorm(n = 100, mean = 0, sd = 1)

#lets plot a histogram of that
hist(norm_numbers)
```



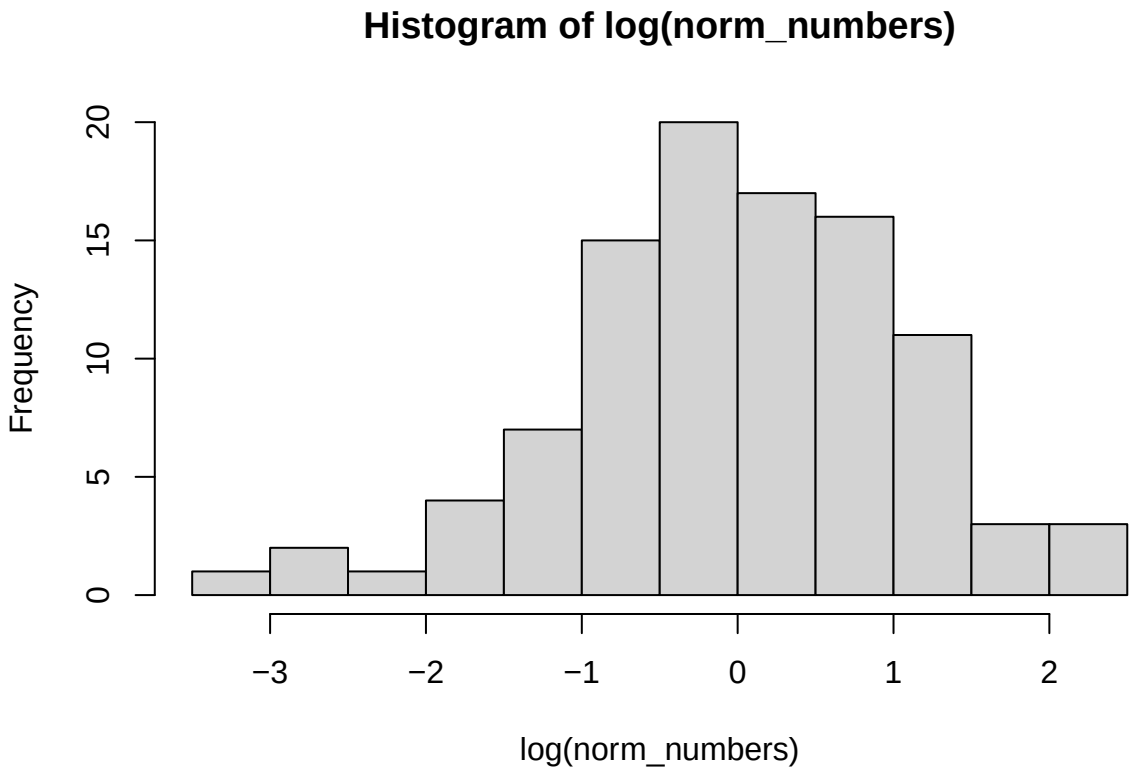
Likewise we can create a lognormal distribution using rlnorm()

```
#generate values for a normal distribution
norm_numbers <- rlnorm(n = 100, meanlog = 0, sdlog = 1)

#lets plot a histogram of that
hist(norm_numbers)
```



As this is a log normal distribution we can log it and it should become normal.

```
#lets plot a histogram of the log of a log normal
hist(log(norm_numbers))
```



Using these types of simulations can be very useful for creating datasets to compare real observations to see how differnt they are to some pre expected distribution.