Exercises and Additional Lecture Notes for Statistics for Biology - FS 2021

Stephan Peischl

Table of Contents

# Introduction

These notes should help you get started with the statistics software R, guide you through the practical part of the course and provide additional information on selected topics. Before you continue reading, make sure you have R and R Studio installed on your computer. Try to use R as an calculator as shown in the lecture slides and try to get yourself accustomed to the different windows in the R Studio environment. No prior knowledge about R is necessary and once everything is up and running you can dive right in.

We start by loading some relevant packages for this course. An R package is a collection of functions, data, and documentation that extends the capabilities of base R.

# make sure you have th packages installed using   
# install.packages(packagename)  
  
library(tidyverse)  
library(Rmisc)  
library(tibble)  
library(dplyr)  
  
  
# here i define a color palette to use as standard when plotting with ggplot  
# this is purely for aesthetic reasons   
  
cbp2 <- c("#000000", "#E69F00", "#56B4E9", "#009E73",  
 "#F0E442", "#0072B2", "#D55E00", "#CC79A7")  
  
scale\_colour\_discrete <- function(...) {  
 scale\_colour\_manual(..., values = cbp2)  
}

# Visualizations with ggplot2 and base R

## GGplot 2

We start with some examples of how to visulaize data quickly and easily. This should show you how to produce high quality visulaizations using R and motivate you to learn more about programming and data analysis with R.

R has several systems for making graphs, and I have chosen to start with ggplot2 beacuse it is elegant, versatile and is easy to leanr if you have no prior knowledge about programming. For those of you who are already familiar wiht programming langauges, doing graphics in “base R” may seem more intuitive initially but I hope I can convince you that the grammar of plotting used by ggplot has lots of advantages and is very readable once you get the hang of it.

If you like to learn more about the theoretical underpinnings of ggplot2 before you start, I would recommend reading “The Layered Grammar of Graphics”, <http://vita.had.co.nz/papers/layered-grammar.pdf>.

### The MPG dataset

Let us use our first graph to answer a question: Do cars with big engines use more fuel than cars with small engines? What does the relationship between engine size and fuel efficiency look like? Is it positive? Negative? Linear? Nonlinear?

We can find an answer with the mpg data set found in ggplot2 (aka ggplot2::mpg). A data frame is the R data format to store data in a spreadsheet: a rectangular collection of variables (in the columns) and observations (in the rows). The data frame mpg contains observations collected by the US Environmental Protection Agency on 38 models of car.

# the command head() shos us the first few rows of the data set to   
# get an overview of what is stored in our dataset  
  
head(mpg)

## # A tibble: 6 x 11  
## manufacturer model displ year cyl trans drv cty hwy fl class   
## <chr> <chr> <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>   
## 1 audi a4 1.8 1999 4 auto(l5) f 18 29 p compa…  
## 2 audi a4 1.8 1999 4 manual(m5) f 21 29 p compa…  
## 3 audi a4 2 2008 4 manual(m6) f 20 31 p compa…  
## 4 audi a4 2 2008 4 auto(av) f 21 30 p compa…  
## 5 audi a4 2.8 1999 6 auto(l5) f 16 26 p compa…  
## 6 audi a4 2.8 1999 6 manual(m5) f 18 26 p compa…

Among the variables in mpg are:

**displ**, a car’s engine size, in litres.

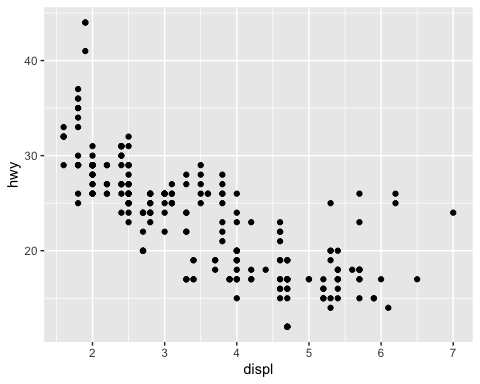
**hwy**, a car’s fuel efficiency on the highway, in miles per gallon (mpg). A car with a low fuel efficiency consumes more fuel than a car with a high fuel efficiency when they travel the same distance.

To learn more about mpg, open its help page by running ?mpg. in the next seciton we learn how to visualize such data.

### The grammar of ggplot2

In general you begin a plot with the function *ggplot()*. ggplot() creates a coordinate system that you can add layers to. The first argument of ggplot() is the dataset to use in the graph. For instance, typing ggplot(data = mpg) into the console creates an empty graph (which is boring so it is not shown here). You can now add layers to your plot. For example, the function geom\_point() adds a layer of points to your plot, thus creating a scatterplot.

ggplot(data = mpg) +   
 geom\_point(mapping = aes(x = displ, y = hwy))

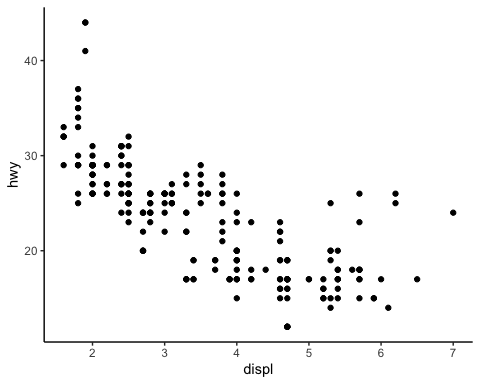


There are many other such *geometric functions* withing the ggplot2 universe and you can combine them in a single plot. You will learn a whole bunch of them throughout this course, but be aware that we will only scratch the surface of what is available. After this course, you should be ready to equip your self with new tools for data analysis whenever you need them.

Each geom function in ggplot2 takes a mapping argument. This defines how variables in your dataset are mapped to visual properties. The mapping argument is always paired with aes(), and the x and y arguments of aes() specify which variables to map to the x and y axes. ggplot2 looks for the mapped variables in the data argument, in this case, mpg.

Next, we use a different theme beacuse this gray background in the standard design is ugly.

ggplot(data = mpg) +   
 geom\_point(mapping = aes(x = displ, y = hwy)) +  
 theme\_classic()



The general grammar of ggplot is

ggplot(data = <DATA>) +

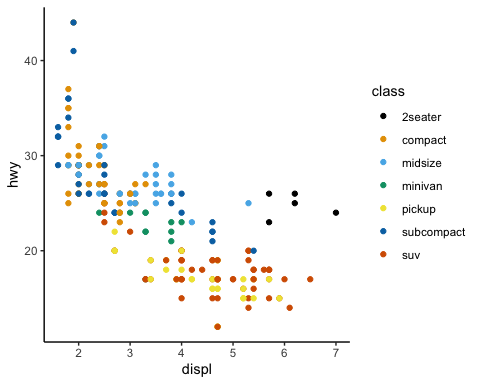
<GEOM\_FUNCTION>(mapping = aes(<MAPPINGS>))

We will explain each part step by step in examples. The DATA argument should be quite clear: here you specify the dataframe (we will learn more about that alter) that you want to work with.

The second part, GEOM\_FUNCTION, specifies what kind of plot you want. As mentioned above *geom\_point* gives you a classic scatter plot. Many other such funtions exist and we will encounter some of them.

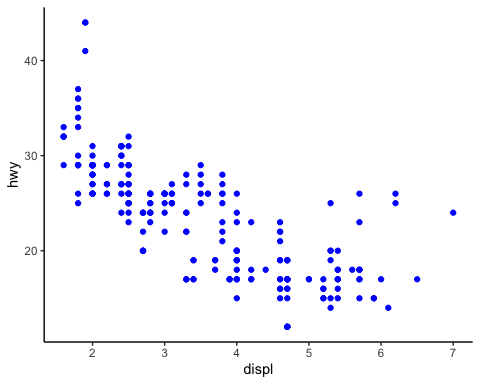
Finally, MAPPING specifies which variables hould be plotted. We ussually specify this with an aesthetics mapping of the form *mapping = aes(x = …, y =…)* to deterimne the variables that give us the x- and y-coordinates. We can however also change the aesthetics of our plot in various way, for instance by giving data points different colors that indicate the value of a third variable:

ggplot(data = mpg) +   
 geom\_point(mapping = aes(x = displ, y = hwy,color = class)) +  
 theme\_classic()



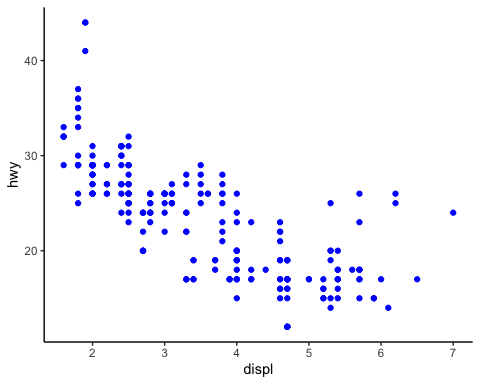
Another important option is to set a clor manually:

ggplot(data = mpg) +   
 geom\_point(mapping = aes(x = displ, y = hwy), color = "blue") +  
 theme\_classic()

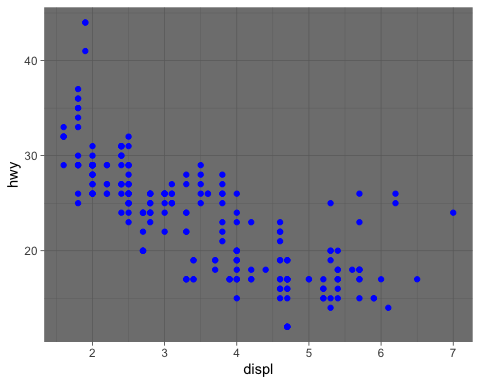


As a final remark, you can also create a plot, store it in a variable and then explore themes after that:

p = ggplot(data = mpg) +   
 geom\_point(mapping = aes(x = displ, y = hwy), color = "blue")   
  
p+theme\_classic()



p+theme\_dark()



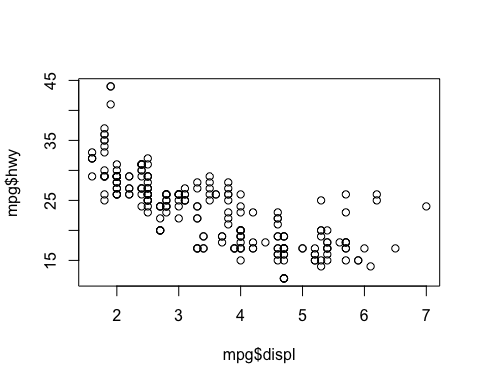
We now set our prefered them globally, so we do not need to specifiy it all the time:

theme\_set(theme\_classic())

## Base R graphics

Base R refers to the standard R functions. While ggplot allows us to make sophisticated graphs very quickly, I find that in some situations base R is just as good or even better. Let us recreate the same plot as before:

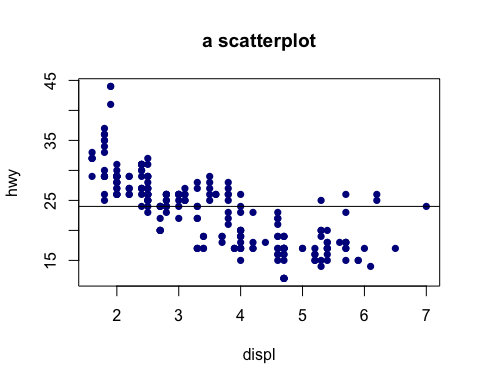
plot(mpg$displ,mpg$hwy)



Note how we use the $ symbol to access the variables stored in a data frame here; it allows you to access a specific column of a data frame. Try typing the name of a dataframe, followed by the $ operator in RStudio; it should automatically suggest you the availbale variable names in your data frame.

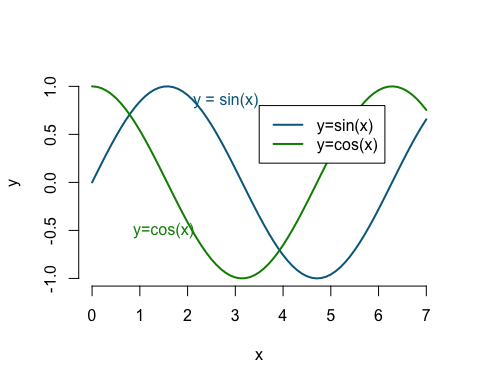
We can alos add stuff to this plot, change colors, and pretty much everything else we would like to. A quick look at the help function should give you an overview about the scope of the function plot and it’s parameters. For instance, have a look at the following plot, where I make a scatterplot and modifiy the appearance of the symbols, add some labels and a title, and then add a horizontal line at the value of the median fuel efficiency:

plot(mpg$displ,mpg$hwy,pch=16,col="darkblue",xlab="displ",ylab="hwy", main ="a scatterplot")  
abline(h = median(mpg$hwy))



When working with data frames, ggplot may seem more attractive and easy to use than base R functions, simply because many steps are automated. You may want to try color-coding the different points according to the varibale class, just as we did before. Then try to add a legend (look up the function *legend* in the R help). You will find out that it may be a bit tedious to put all this things together. However, it also gives you a lot of control about the final looks of your figure once you have figured out what you want and how you get it. Another advantge of base R functions is that you are not restircted to the data frame format, which often makes things easier. For instance, consider this plot:

x=seq(0,7,by=0.1)   
# x = (0,0.1,0.2,0.3, ... ,6.9,7)  
y=sin(x)  
  
  
plot(x,y,lwd=2,type="l",col="deepskyblue4",axes=FALSE)   
#plot without axes  
  
axis(1) #add axes to plot  
axis(2)  
  
text(2.8,0.85,"y = sin(x)",col="deepskyblue4")   
# add some text at coordinates x = 2.8, y = 0.85  
  
lines(x,cos(x),lwd=2,col="green4") # add a cosine plot   
text(1.5,-0.5,"y=cos(x)",col="green4") # and label it  
  
legend(3.5,0.8,legend=c("y=sin(x)","y=cos(x)"),  
col=c("deepskyblue4","green4"),lwd=2)



# add a legend to the plot

Instead of using a data frame, we have created vectors of variables and used them in several different ways. You will explore this more in the exercise sessions.

By now you should see that R is very versatile and can be used for much more than simply doing statistics. This course will teach you the basics of R so that you can start using it for your research projects. An essential part is that you should also learn how to find and use additional resources, such as the help function in R, books, the internet, etc. Nobody knows all of R - it is an ever growing community endevaour and looking up how things work, which functions or packages are available, or how other people solved a problem is an essential part of learning to be proficient in R.

# Working with data frames

## Loading datasets

How to we feed R our data? Of course, we can just type it into the console, a bit like in the previous base R graphics example. For large data sets (or rather anythign that is not an eytremely small data set) this is not practical. We want to load our data from a file, which makes our analysis easily reproducible if we share data and R scripts with someone. First we need to make sure R knows where our working directory is, using the function *setwd* (set working directory):

setwd("/Users/stephan/Dropbox/Teaching/StatisticsForBiology/")

If you want to check your current working directory, you can do this with *getwd()*.

I use a dataset about the number of plates on sticklebacks. You can find this dataset (and many others) on ilias. We can load a dataset using the function *read.csv* and store it in the variable stickleback

stickleback = read.csv("chap03e3SticklebackPlates.csv")

The variable *stickleback* now contains a so called data frame. We can have a quick summary of the content of this data frame using the command *str* (short for *STR*ucture):

str(stickleback)

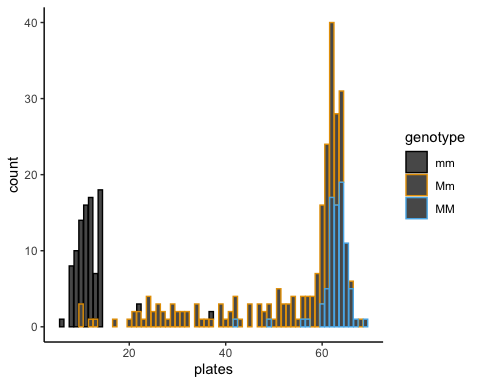
## 'data.frame': 344 obs. of 3 variables:  
## $ id : Factor w/ 344 levels "4-1","4-10","4-100",..: 1 102 282 293 2 22 42 131 212 280 ...  
## $ plates : int 11 63 22 10 14 11 58 36 31 61 ...  
## $ genotype: Factor w/ 3 levels "mm","Mm","MM": 1 2 2 2 1 1 2 2 2 2 ...

We can see form the output that we have 344 observations (= sample size) and for each individual we have measured 3 variables (id, plates and genotypes). ID and genotpye are so-called factors, that is, a catergorical variable and plates is of type integer.

We used the function *read.csv* because our data is stored in the CSV format (short for *c*omma *s*eperated *v*alues - open the file in a text editor to see what that means). CSV is a common format and you can export your data from software like Excel in that format. Of course, R provides functions for other formats as well or you an use the read.table() function where you can specify how the data should be read and transformed into a data frame.

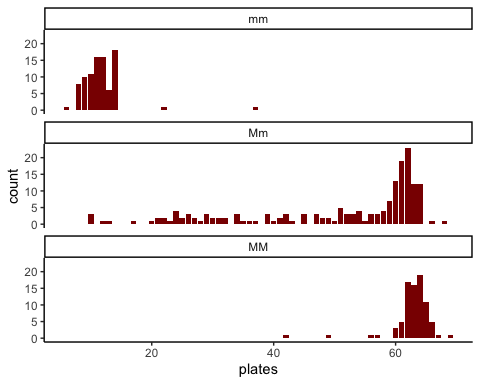
Let us use our ggplot skills to create a nice figure that visualizes our data set. For this we introduce the fucntion *geom\_bar()* which allows us to make a bar plot of the number of plates per indivudal, colored by genotype:

ggplot(data = stickleback) +   
 geom\_bar(mapping = aes(x = plates,color=genotype)) +   
 theme\_classic()



This figure is a bit messy. A better way would be to show the distribution for each genotype in its own window. This can be done using the function *facet\_wrap*. As before, we can simply add this to our ggplot command:

ggplot(data = stickleback) +   
 geom\_bar(mapping = aes(x = plates),fill="darkred") +   
 facet\_wrap(~genotype,nrow = 3) +  
 theme\_classic()



We used a new symbol here: the ~. This is part of an R object called formula and it will alter become clear why we used it here like this. For now, just remember to use it in front of the varibale name in facet wrap. The second aprameter is simply the number of rows in the plot. Try changing it and see what happens. Note: the variable that you pass to facet\_wrap shoudl **always** be categorical!

## Manipulating data frames

A data frame is essentially a spreadsheet or a table (or rather a matrix). Therfore, we acess each column or each row, if we want to. This can be done using the *[,]* symbols. The element in row 3 and column 5 is given by:

stickleback[3,5]

## NULL

the whole 3rd column can be accesses by

stickleback[,3]

## [1] mm Mm Mm Mm mm mm Mm Mm Mm Mm Mm mm mm mm Mm Mm mm Mm Mm MM Mm Mm mm Mm MM  
## [26] Mm mm MM Mm Mm MM mm Mm Mm mm Mm mm MM Mm mm Mm MM mm Mm Mm MM Mm mm mm Mm  
## [51] Mm MM mm mm mm Mm MM MM Mm MM MM Mm MM Mm mm mm MM Mm Mm MM MM Mm MM Mm MM  
## [76] MM Mm Mm Mm mm Mm Mm Mm Mm MM Mm Mm Mm Mm MM mm Mm Mm mm MM Mm mm MM mm mm  
## [101] Mm mm mm Mm MM Mm mm mm Mm Mm MM MM Mm mm Mm MM MM Mm mm Mm Mm MM mm MM mm  
## [126] mm Mm Mm Mm mm Mm mm Mm mm Mm MM Mm Mm Mm Mm MM Mm MM Mm Mm Mm mm MM Mm MM  
## [151] MM MM Mm Mm mm mm Mm MM MM Mm Mm mm Mm Mm mm Mm mm Mm Mm MM Mm MM Mm MM MM  
## [176] Mm mm MM mm Mm Mm Mm Mm Mm Mm Mm mm Mm MM Mm Mm MM mm MM Mm mm Mm Mm Mm Mm  
## [201] Mm Mm Mm Mm Mm Mm MM MM Mm Mm mm Mm mm mm Mm MM Mm Mm Mm Mm mm mm Mm Mm Mm  
## [226] mm MM MM MM Mm Mm mm MM MM MM MM Mm Mm MM Mm MM mm mm Mm Mm Mm MM mm mm MM  
## [251] mm Mm MM Mm Mm mm mm Mm Mm MM mm Mm Mm mm Mm Mm Mm Mm mm MM Mm mm Mm MM Mm  
## [276] mm Mm Mm MM mm mm Mm MM Mm Mm mm MM mm Mm Mm Mm MM mm MM Mm Mm mm MM MM mm  
## [301] MM MM MM Mm Mm MM mm mm MM MM mm Mm Mm Mm Mm MM mm Mm Mm Mm Mm MM Mm mm Mm  
## [326] Mm Mm Mm Mm Mm mm MM mm mm Mm Mm Mm mm Mm mm MM mm Mm mm  
## Levels: mm Mm MM

and the 5th row by

stickleback[5,]

## id plates genotype  
## 5 4-10 14 mm

Note that stickleback[[,3]] gives you all measurment of the 3rd variable in our dataframe and is hence equivalent to stickleback$genotype. The vector stickleback[5,] on the other hand gives you a vector with the 3 measured values of the 5 th individual.

We can also use this way of accessing elements to find elements of a certain type, for instance, fi we want to know the genotype of all individuals with more then 30 plates, we can get this in the following way:

stickleback[stickleback$plates>20,3]

Try to understand what is happening here by teasing apart and investigating the differnt parts of this (nested) command. What would happen if you removed the *stickleback$plates>20* from the command, or the *3*.

A slightly more elegant way to get this is by using the function *filter* (make sure you have the package dyplr loaded!):

filter(stickleback,plates>20)

which gives ou a data frame with only the individuals that satisfy the condition set in the argument.

## Adding variables to data frames

Sometimes we wish to extend our dataframe with calcualtions we did during our analysis or with additional measurements. This can be done with the function *mutate()*. It simply adds new columns at the end of your dataset. Let’s say we want to identify which indivduals have fewer or more plates as comapred to the global mean:

diff.to.mean = stickleback$plates - mean(stickleback$plates)  
stickleback2 = mutate(stickleback,difference = diff.to.mean)  
head(stickleback2)

## id plates genotype difference  
## 1 4-1 11 mm -32.43314  
## 2 4-2 63 Mm 19.56686  
## 3 4-4 22 Mm -21.43314  
## 4 4-5 10 Mm -33.43314  
## 5 4-10 14 mm -29.43314  
## 6 4-12 11 mm -32.43314

## Subsetting a data frame

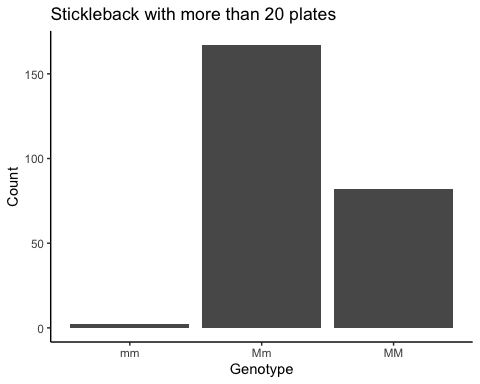
We have alreaddy seen how to look at a subset of your data frame using indexing. Here is another slightly more elegant way of doing this, especially when many conditions are combined over multiple variables. For instance, what if we want to consider only stickleback with more than 20 plates:

head(subset(stickleback, plates>20))

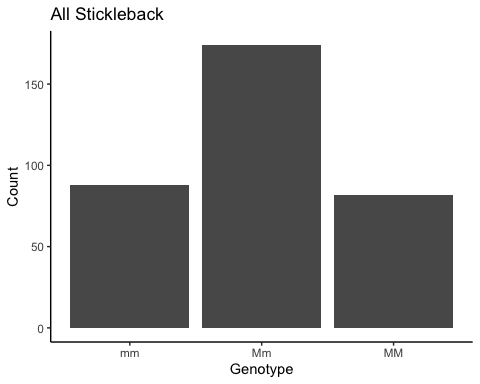
## id plates genotype  
## 2 4-2 63 Mm  
## 3 4-4 22 Mm  
## 7 4-14 58 Mm  
## 8 4-23 36 Mm  
## 9 4-31 31 Mm  
## 10 4-38 61 Mm

We could now compare how the disitrbution of genotypes changes if we condition on more than 10 plates:

ggplot(data = subset(stickleback, plates>20)) +   
 geom\_bar(mapping= aes(x = genotype)) +  
 labs(title="Stickleback with more than 20 plates",  
 x ="Genotype", y = "Count")



ggplot(data = stickleback) +   
 geom\_bar(mapping= aes(x = genotype)) +  
 labs(title="All Stickleback",  
 x ="Genotype", y = "Count")

 There are a few *mm* genotypes with more than 20 plates. But how many?

subset(stickleback,plates > 20 & genotype == "mm")

## id plates genotype  
## 100 4-25 37 mm  
## 132 4-87 22 mm

It turns out there are exactly two: one with 37 plates and one with 22. We have used logical operators here: “&” means that both conditions need to be satisfied. In contrast, try to see what the symbol " | " means:

head(subset(stickleback,plates > 20 | genotype == "mm"))

## id plates genotype  
## 1 4-1 11 mm  
## 2 4-2 63 Mm  
## 3 4-4 22 Mm  
## 5 4-10 14 mm  
## 6 4-12 11 mm  
## 7 4-14 58 Mm

You may have guessed it already, the " | " represetns the logical “or” command, indicating that either of the two conditions needs to be satisfied. Note that for logical comparions we use the “==” and not the “=” symbol, which is reserved for assignments. What is the difference? Try it:

a = 5  
  
a == 5

## [1] TRUE

a == 6

## [1] FALSE

We can also use the symbols < (smaller), > (larger), <= (smaller or equal), >= (larger or equal), != (not equal) and combine them into logical statements.

### *Practice Questions:*

* Add a new varaible to the stickleback data frame that contains the difference between plate number and the mean plate number for that individuals genotype.
* Load the flights dataset:

library(nycflights13)

Find all flights that

* Had an arrival delay of two or more hours
* Flew to Houston (IAH or HOU)
* Were operated by United, American, or Delta
* Departed in summer (July, August, and September)
* Arrived more than two hours late, but didn’t leave late
* Were delayed by at least an hour, but made up over 30 minutes in flight
* Departed between midnight and 6am

# Calculating Descriptive Statistics

We continue using the stickleback data set and calcualte some statistics on it. A simple way to get a quick overview about the properties of your data frame is using the funciton *summary*:

summary(stickleback)

## id plates genotype  
## 4-1 : 1 Min. : 6.00 mm: 88   
## 4-10 : 1 1st Qu.:14.00 Mm:174   
## 4-100 : 1 Median :57.00 MM: 82   
## 4-101 : 1 Mean :43.43   
## 4-103 : 1 3rd Qu.:62.00   
## 4-105 : 1 Max. :69.00   
## (Other):338

If you want to calcualte a specific statistic, like the mean of a variable, this can be done easily:

mean(stickleback$plates)

## [1] 43.43314

You may have noticed that we have no calcualted the mean number of plates for *ALL* fish in our data set. What if we want to calcualte the number of plates for a specific genotype? We can use the function filter:

stickleback.mm = filter(stickleback,genotype=="mm")  
mean(stickleback.mm$plates)

## [1] 11.67045

You could now calcualte the mean plate numbers for each genotpye in this way:

mean.mm = mean((filter(stickleback,genotype=="mm"))$plates)  
mean.mM = mean((filter(stickleback,genotype=="mM"))$plates)  
mean.MM = mean((filter(stickleback,genotype=="MM"))$plates)

**Note:** The tidyverse offers an elegant way to write a long series of commands in a very readable format. To show you how to combine multiple commands quickly, we first need two more functions, *group\_by* and *summarize*, and a new operator called the pipe: %>% (because we create a pipeline thorugh which our data “flows”). The above code to calculate the mean for each genotype would become:

stickleback %>%   
group\_by(genotype) %>%  
summarise(avg = mean(plates))

## avg  
## 1 43.43314

This is very readable: you take the dataframe stickleback, group it by genotye and summarize it by calcualting the mean number of plates. However, errors might be more difficutl to spot when the code is written in that way, as intermediate steps cannot be checked.

In the exercises you will learn a few more useful functions to rearrange, filter, extend and edit data frames.

*Practice Question:* Can you come up with anotehr way to do this whithout using filter?

*Two possible answers (there are many ways to do this):* Straightforward “indexing”:

mean.mm = mean(stickleback$plates[stickleback$genotype=="mm"])  
mean.mM = mean(stickleback$plates[stickleback$genotype=="mM"])  
mean.MM = mean(stickleback$plates[stickleback$genotype=="MM"])

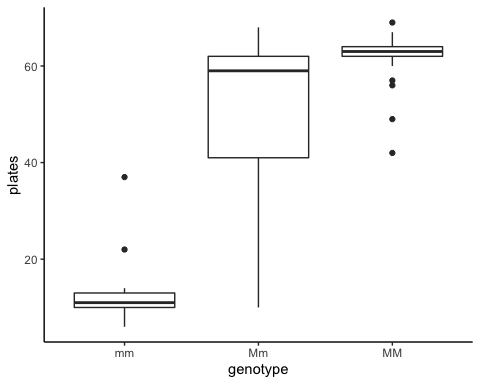
or by using the function *tapply* (look up the help page for this function by typing “?tapply”):

mean.plates = tapply(stickleback$plates,stickleback$genotyp,mean)

## Plots

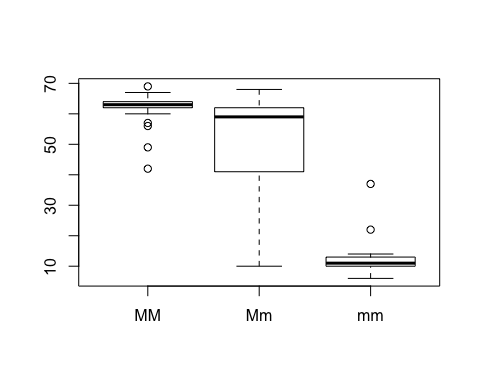
We can do a boxplot easily using ggplot:

ggplot(data = stickleback) +   
 geom\_boxplot(mapping = aes(x = genotype, y = plates))



In base R, this can be done as well:

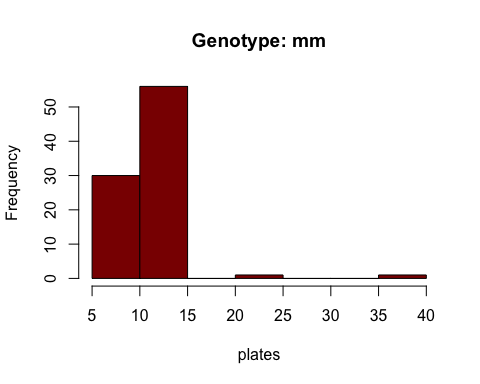
boxplot(stickleback$plates[stickleback$genotype=="MM"],  
 stickleback$plates[stickleback$genotype=="Mm"],  
 stickleback$plates[stickleback$genotype=="mm"],  
names = c("MM","Mm","mm")  
)



*Practice Question:* Can you plot a histogram of plate numbers of individuals with genotpye mm using base R plotting functions? Hint: Use what you have learned here and look up the function hist(). Alternatively, use ggplot and geom\_histogram.

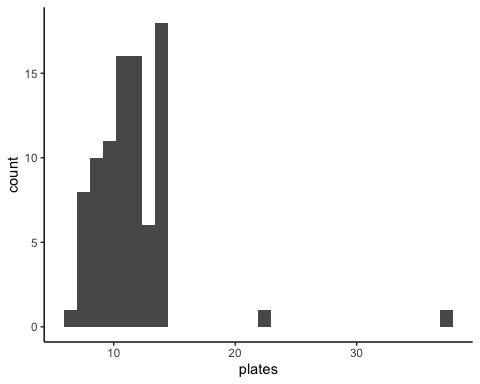
*Answer:*

hist(stickleback$plates[stickleback$genotype=="mm"],col="darkred",xlab="plates",main="Genotype: mm")



ggplot(data=filter(stickleback,genotype == "mm")) +   
 geom\_histogram(mapping = aes(x = plates))

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



# A bit of workflow

In this part you will see how to explore a data set and store your analysis in a reproducible and reusable way.

## What is an R script

So far we only used the console of R. We enter commands and R executes them. Most of the time we want to have a whole bunch of commands stored in a single space: loading the file, preparing and cleaning the data,make figures, etc. The solution is easy, we jsut store our commands in a text file with the extension *.R* and tell R do execute this script. Evene better, RStudio has its own window panel for writing and executing scripts! Try to open a new R script (via File -> New -> R script) and just write your code in the newly opened window. When you have written your code, you can eitehr execute it step by step by putting the mouse cursor in the line you want to execute and then hit then *Run* button. If you want to eecute the whole script from beginning to end, just hit the *Source* button. This allows you to make reproducible scripts than you can modify and reuse whenever you need to. This is probably the biggest advantage of R. Once you have solved a problem and wrote a script for it, you can always reuse it. The more you use R, the less work will be necessary for each new project!

In the next part I will introduce the basic tools for creating an automated analysis that consists of several steps.

## Defining and using Variables

Defining variables is really easy in R. Unlike in other programming languages we do not need to precisly define what kind of variable we want. We can just werite down a name and assign a value, R figures out the rest. For instance:

x = 5

automatically creates a *numeric* variable and assigns the value 5 to it. This command

vec = c(1,2,3,4,5,6)

creates a vectors that contains the values 1 to 6. The *c* here stands for *concatenate* and tells us to combine all the numbers into a single vector. You can access the elements of the vector by their index, e.g., we can set the 5th element of the vector to 0 in the follwoing way:

vec[5] = 0

In general it is advised to define the type of a varible before using it, and also tell R how big this variable will be (that is, how much memory we need to reserve to store it). For instance, if we want to define a 3x4 matrix we can use the command *matrix*:

mat=matrix(ncol=3,nrow=4)

You can then enter numbers in the rows and columns using the [] operator:

mat

## [,1] [,2] [,3]  
## [1,] NA NA NA  
## [2,] NA NA NA  
## [3,] NA NA NA  
## [4,] NA NA NA

mat[2,3]= 5   
mat

## [,1] [,2] [,3]  
## [1,] NA NA NA  
## [2,] NA NA 5  
## [3,] NA NA NA  
## [4,] NA NA NA

You can see that the matrix is intially filled with the object *NA* which stands for *not available* and ist the term R uses for missing data. Just like in dataframes (after all a data frame is more or less a matrix with benefits), you can also name columns and rows and use them to access the elements:

mat = matrix(1:25,nrow=5,ncol=5)  
colnames(mat) = c("var1","var2","var3","var4","var5")  
rownames(mat) = c("obs1","obs2","obs3","obs4","obs5")  
mat

## var1 var2 var3 var4 var5  
## obs1 1 6 11 16 21  
## obs2 2 7 12 17 22  
## obs3 3 8 13 18 23  
## obs4 4 9 14 19 24  
## obs5 5 10 15 20 25

mat["obs2",]

## var1 var2 var3 var4 var5   
## 2 7 12 17 22

mat[,"var3"]

## obs1 obs2 obs3 obs4 obs5   
## 11 12 13 14 15

mat["obs2","var3"]

## [1] 12

If we want to store some text, we can write

word = "hello"

and R will automatically recognize that the string of letters “hello” is not numeric and will creat a variable of type *character*. Try to see how this is different from

letters = c("h","e","l","l","o")

Next try to see what happens here:

hello = 10  
word1 = "hello"  
word2 = hello  
print(word1)

## [1] "hello"

print(word2)

## [1] 10

## Loops

Before we have seen how to calcualte the mean number of plates for each genotype. Essentially we have done the same thing 3 times. Imagine we woul dhave hundreeds of genotypes. In such situations loops come in handy. They allow us to automate a process that needs to be repeated many times. We only introduce the so-called *for-loop* here, which will allow you to do a lot of things. Once you have understood how it works, it will be easy to figure out how a *while-loop* works and what the differences are.

Here is a simple for loop that calcualtes the mean number of plates for each genotpye in our stickleback data set:

# first we create a vector that contains all genotypes in our dataset  
# we can do this manually:   
# genotypes = c("mm","mM","MM")  
# or by using the function levels()  
# that gives us a vector with all the different entries found in a   
# vector of type factor  
  
genotypes = levels(stickleback$genotype)  
  
# next, we define a vector that has the same length as the number   
# of different genotypes and name the elements accordingly  
  
plate.numbers = vector("numeric",length(genotypes))   
names(plate.numbers)=genotypes  
  
# no comes the actual loop:  
for(g in genotypes)  
{  
 plate.numbers[g]=mean(stickleback$plates[stickleback$genotype==g])  
}

We can see what is happening here by showing the intermediate steps more explicitly:

i = 1  
for(g in genotypes)  
{  
 print(paste("This is iteration ",i," of our for-loop"))  
 print(paste("Genotype ",i," = ",g))  
 mean.temp = mean(stickleback$plates[stickleback$genotype==g])  
 print(paste("The mean number of plates of genotype ",g," is ",mean.temp))  
 plate.numbers[g]=mean.temp  
 i = i+1  
}

## [1] "This is iteration 1 of our for-loop"  
## [1] "Genotype 1 = mm"  
## [1] "The mean number of plates of genotype mm is 11.6704545454545"  
## [1] "This is iteration 2 of our for-loop"  
## [1] "Genotype 2 = Mm"  
## [1] "The mean number of plates of genotype Mm is 50.3793103448276"  
## [1] "This is iteration 3 of our for-loop"  
## [1] "Genotype 3 = MM"  
## [1] "The mean number of plates of genotype MM is 62.780487804878"

There are 3 basics steps here:

* Reserve memory for the output
* Set up how often the loop should be iterated
* The code that should be repeatedly calcualted

*Question: Could you come up with an alterantive way to solve this problem using a loop?*

## If / else

Another useful tool are if/else constructs. They allow you to make choices during your script. Let us again consider the sticklbeack data set. Let say we want to calcualte the difference between the number of plates of plates of an individual and the mean of that indivudal’s genotype. We can do this by combining loops with if/else statements (admittedly, there are better ways to achieve this but it is a good instuctive example):

number.obs = dim(stickleback)[1]  
diff.to.mean = vector("numeric",length=number.obs)  
for (i in 1:number.obs)  
{  
 if(stickleback$genotype[i] == "mm")  
 diff.to.mean[i] = stickleback$plate[i] - mean.mm  
   
 if(stickleback$genotype[i] == "mM")  
 diff.to.mean = stickleback$plate[i] - mean.mM  
   
 if(stickleback$genotype[i] == "MM")  
 diff.to.mean = stickleback$plate[i] - mean.MM  
}  
  
  
stickleback3= mutate(stickleback2,diff.per.genotype = diff.to.mean)  
  
head(stickleback3)

## id plates genotype difference diff.per.genotype  
## 1 4-1 11 mm -32.43314 -0.7804878  
## 2 4-2 63 Mm 19.56686 NA  
## 3 4-4 22 Mm -21.43314 NA  
## 4 4-5 10 Mm -33.43314 NA  
## 5 4-10 14 mm -29.43314 NA  
## 6 4-12 11 mm -32.43314 NA

# Working with Distributions and random variables

## Functions to work with probaility distributions

R has a set of functions to work with distirbutions and random numbres. Have a look at

help(Distributions)

if you want to know more. Let us focus on the normal distirbution. It’s density is given by

The function *dnorm* returns the value of the probability density function for the normal distribution for x, given parameters and . Some examples of using dnorm are below:

# We use the pdf of the normal with x = 0,   
# mu = 0 and sigma = 0.   
# The dnorm function takes three main arguments,   
# as do all of the \*norm functions in R.  
  
dnorm(0, mean = 0, sd = 1)

## [1] 0.3989423

# Another exmaple of dnorm where parameters have been changed.  
  
dnorm(2, mean = 5, sd = 3)

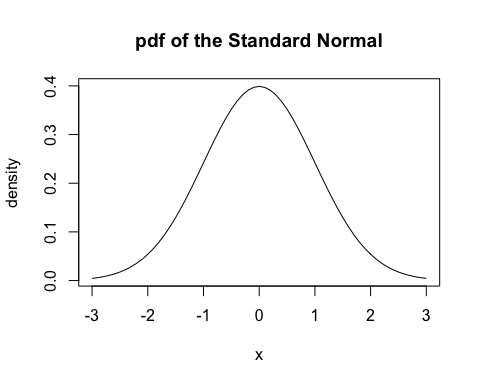
## [1] 0.08065691

Let us do a plot showing the density fucntion of the normal distirbution. We use base R as ggplot works fine with data frames and here we prefer to wrok with vectors instead.

# The plot should show x values on the x-axis and the dnorm(x,...)   
# values on the y-axis  
  
# First I'll make a vector of x values  
x\_vals <- seq(-3, 3, by = .1)  
  
# Let's have a look  
x\_vals

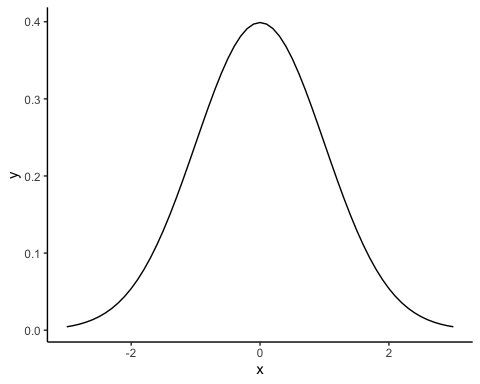
## [1] -3.0 -2.9 -2.8 -2.7 -2.6 -2.5 -2.4 -2.3 -2.2 -2.1 -2.0 -1.9 -1.8 -1.7 -1.6  
## [16] -1.5 -1.4 -1.3 -1.2 -1.1 -1.0 -0.9 -0.8 -0.7 -0.6 -0.5 -0.4 -0.3 -0.2 -0.1  
## [31] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4  
## [46] 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9  
## [61] 3.0

# The y values are then given by dnorm  
y\_vals <- dnorm(x\_vals,0,1)  
  
# Now we'll plot these values  
plot(x\_vals,y\_vals,   
 type = "l", # Make it a line plot  
 main = "pdf of the Standard Normal",  
 xlab= "x",ylab="density")



Of course, you could create a data frame with these values and then use ggplot if you want to:

std\_norm = data.frame(x= x\_vals,y=y\_vals)  
  
ggplot(data = std\_norm) +  
 geom\_line(mapping = aes(x = x,y=y))



The other three most relevant functions are: *pnorm* (cumulative distribution function), *qnorm* (quantiles) and *rnorm* (draw random numbers). The ue of *pnorm* is completely analogous to *dnorm*, so we skip it here and focus on themore interesting *qnorm* and *rnorm*.

## Quantiles

The function qnorm gives us the quantiles of the normal distirbution:

# Let's make a vector of :   
# from 0 to 1 by increments of .05  
quantiles <- seq(0, 1, by = .05)  
quantiles

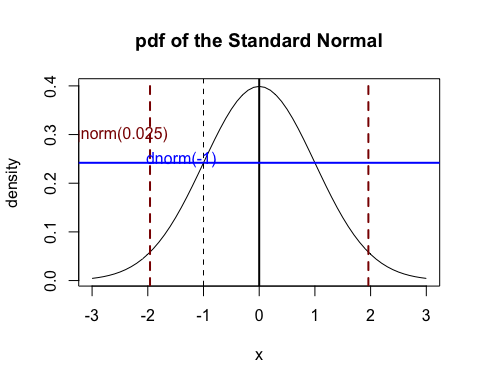
## [1] 0.00 0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40 0.45 0.50 0.55 0.60 0.65 0.70  
## [16] 0.75 0.80 0.85 0.90 0.95 1.00

# Now we'll find the value for each of those quantiles  
# Remeber that the q-quantile is the value Q such that  
# P(X>Q) = q   
# In other words, the quantile is the inverse of the CDF  
  
qvalues <- qnorm(quantiles)  
qvalues

## [1] -Inf -1.6448536 -1.2815516 -1.0364334 -0.8416212 -0.6744898  
## [7] -0.5244005 -0.3853205 -0.2533471 -0.1256613 0.0000000 0.1256613  
## [13] 0.2533471 0.3853205 0.5244005 0.6744898 0.8416212 1.0364334  
## [19] 1.2815516 1.6448536 Inf

We could now plot the PDF of a normal distribution and illustrate a few of the concepts:

plot(x\_vals,y\_vals,   
 type = "l", # Make it a line plot  
 main = "pdf of the Standard Normal",  
 xlab= "x",ylab="density")   
  
# the 50 % quantile is just the median:  
abline(v = qnorm(0.5),col="black",lwd=2,lty=1)  
  
# the 2.5 % and the 97.5% quantile indicated the area where 95% of the most   
# common data lies:  
abline(v = qnorm(c(0.025,0.975)),col="darkred",lwd=2,lty=2)  
text(-2.5,0.3,"qnorm(0.025)",col="darkred")  
  
# In contrast, the function dnorm gives us the value of the PDF  
# for a specfific x value, that is, the height of the function f(x)   
# at the point x  
  
abline(h = dnorm(-1),col="blue",lwd=2,lty=1)  
abline(v = -1,lty=2)  
text(-1.4,0.25,"dnorm(-1)",col="blue")



## Drawing random numbers

R also allows us to draw random numbers. This is very handy in many situations. The parameters of rnorm are

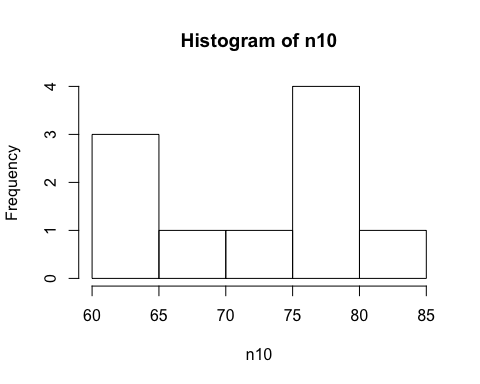
* the number of observations
* the mean of the distirbution
* the standard deviation of the distribution

Here is an example, where we calculate a sample of 0, 100 and 1000 observations of the same distirbution.

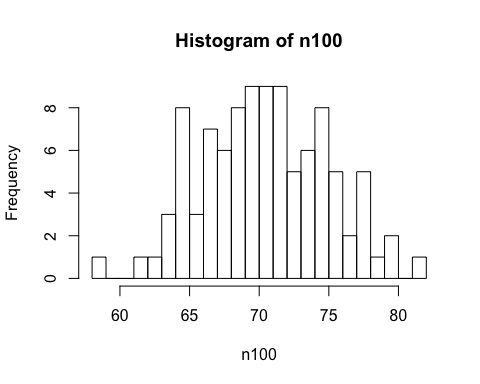
# Let's generate three different vectors of random numbers   
# from a normal distribution  
n10 <- rnorm(10, mean = 70, sd = 5)  
n100 <- rnorm(100, mean = 70, sd = 5)  
n10000 <- rnorm(10000, mean = 70, sd = 5)  
  
# Let's just look at one of the vectors  
n10

## [1] 76.52410 80.84969 64.09680 79.40903 76.17589 64.81172 70.40982 78.05296  
## [9] 60.47464 65.82100

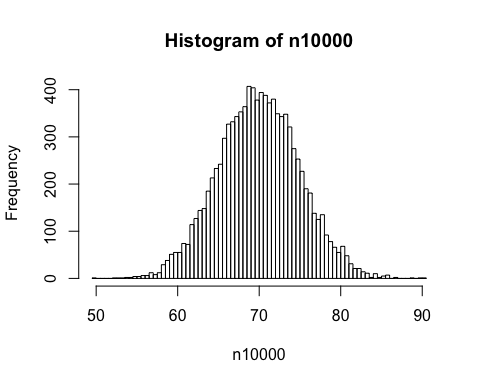
# The breaks argument specifies how many bars are in the histogram  
hist(n10, breaks = 5)



hist(n100, breaks = 20)



hist(n10000, breaks = 100)



*Note:* If we set a *seed* for our random number generator (the algorithm that is used to calcualte a radnom number), we can make our analysis replicable:

set.seed(100)  
rnorm(1,0,1)

## [1] -0.5021924

rnorm(1,0,1)

## [1] 0.1315312

set.seed(100)  
rnorm(1,0,1)

## [1] -0.5021924

rnorm(1,0,1)

## [1] 0.1315312

## Plotting Confidence intervalls

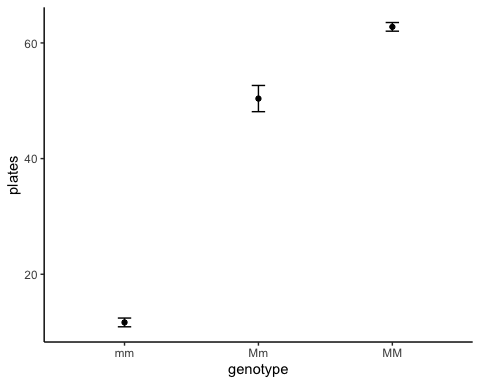
We first generate a data frame with a summary of our data (this can be done in many ways, this is a concise one):

CIs <- summarySE(stickleback, measurevar="plates", groupvars=c("genotype"))  
  
CIs

## genotype N plates sd se ci  
## 1 mm 88 11.67045 3.567805 0.3803293 0.7559456  
## 2 Mm 174 50.37931 15.146866 1.1482809 2.2664440  
## 3 MM 82 62.78049 3.410313 0.3766060 0.7493279

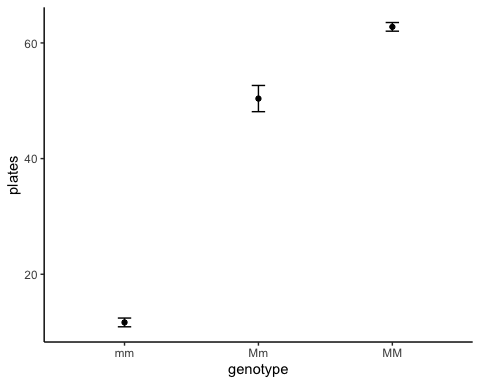
and then we use the data frame with our summary to add error bars to our plot:

ggplot(data = CIs) +  
 geom\_point(mapping = aes(x = genotype,y=plates)) +  
 geom\_errorbar(mapping = aes(x = genotype,y=plates,ymin=plates-ci, ymax=plates+ci), width=.1)



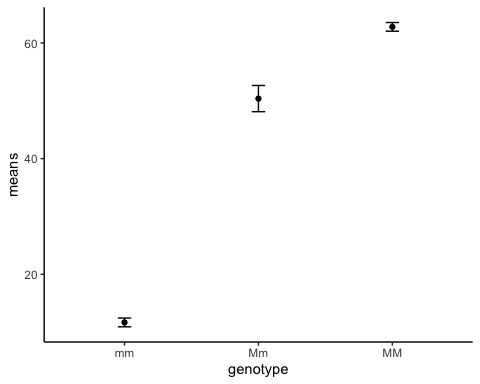
*Note:* One can shorten things by specifying the aesthetics in the ggplot command - this simply means that all geom\_functions use the same x and y variables (which is very often the case, but not always!):

ggplot(data = CIs, aes(x = genotype,y=plates)) +  
 geom\_point() +  
 geom\_errorbar(mapping = aes(ymin=plates-ci, ymax=plates+ci), width=.1)



*Note:* The function *t.test()* can also be used to calcualte confidence intervalls assuming a normal distribution.

genotypes = c("MM","Mm","mm")   
CI.upper = vector("numeric",3)  
CI.lower = vector("numeric",3)  
means = vector("numeric",3)  
  
i = 1  
  
for(g in genotypes)  
{  
   
 results = t.test(stickleback$plates[stickleback$genotype == g])  
   
 CI.lower[i] = results$conf.int[1]  
 CI.upper[i]= results$conf.int[2]  
 means[i] = results$estimate  
 i = i + 1  
}  
  
CI.data = data.frame(genotype = genotypes,mean = means,CI.lower = CI.lower,CI.upper = CI.upper)  
  
ggplot(data = CI.data, aes(x = genotype,y=means)) +  
 geom\_point() +  
 geom\_errorbar(mapping = aes(ymin=CI.lower, ymax=CI.upper), width=.1)



## Putting everything together: A simple simulation

Copy and paste this R code into a script. Before you run it, try to figure out what it does.

reps = 1000  
sample\_size = 10  
mu = 10  
sigma = 5  
means = vector("numeric",reps)  
  
for (i in 1:reps)  
{  
 sample = rnorm(sample\_size,mu,sigma)  
 means[i] = mean(sample)  
}  
  
hist(means,freq=F,main="A distribution")  
  
x = seq(mu-2\*sigma,mu+2\*sigma,by=0.01)  
  
SE = sigma/sqrt(sample\_size)  
  
y = dnorm(x,mu,SE)  
  
lines(x,y)

# Hypothesis Tests

In this chpater we will learn how to do hypotehsis tests in R. This is very simple, as you will see - often a single line of code is sufficient for conducting a test.

## Binomial-test

We use the binomial test to test whether spermatogenesis genes in the mouse genome occur with unusual frequency on the X chromosome. First, we load the dat set and inspect it:

mouseGenes <- read\_csv("~/Dropbox/Teaching/BlogdownPages/StatisticsForBiology/chap07e2SexAndX.csv")  
  
head(mouseGenes)

## # A tibble: 6 x 2  
## chromosome onX   
## <chr> <chr>  
## 1 4 no   
## 2 4 no   
## 3 6 no   
## 4 6 no   
## 5 6 no   
## 6 7 no

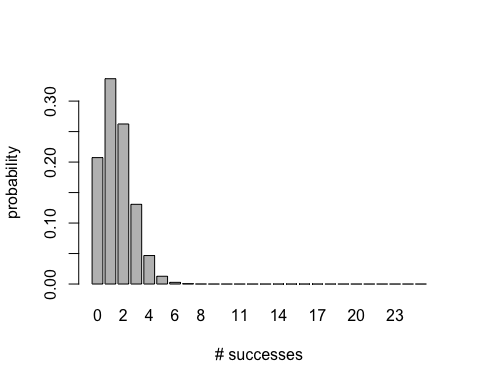
# Tabulate the number of spermatogenesis genes on the   
# X-chromosome and the number not on the X-chromosome.  
  
table(mouseGenes$onX)

##   
## no yes   
## 15 10

# Calculate the binomial probabilities of all possible outcomes  
# under the null hypothesis. Under the binomial  
# distribution with n = 25 and p = 0.061, the number of successes  
# can be any integer between 0 and 25.  
  
xsuccesses <- 0:25  
probx <- dbinom(xsuccesses, size = 25, prob = 0.061)  
data.frame(xsuccesses, probx)

## xsuccesses probx  
## 1 0 2.073193e-01  
## 2 1 3.367007e-01  
## 3 2 2.624760e-01  
## 4 3 1.307255e-01  
## 5 4 4.670757e-02  
## 6 5 1.274386e-02  
## 7 6 2.759585e-03  
## 8 7 4.865905e-04  
## 9 8 7.112305e-05  
## 10 9 8.727323e-06  
## 11 10 9.071211e-07  
## 12 11 8.035781e-08  
## 13 12 6.090306e-09  
## 14 13 3.956429e-10  
## 15 14 2.203032e-11  
## 16 15 1.049510e-12  
## 17 16 4.261188e-14  
## 18 17 1.465509e-15  
## 19 18 4.231266e-17  
## 20 19 1.012696e-18  
## 21 20 1.973624e-20  
## 22 21 3.052667e-22  
## 23 22 3.605629e-24  
## 24 23 3.055193e-26  
## 25 24 1.653947e-28  
## 26 25 4.297797e-31

barplot(probx,names.arg=xsuccesses,xlab="# successes",ylab="probability")



# Use these probabilities to calculate the P-value corresponding to  
# an observed 10 spermatogenesis genes on the X chromosome.   
# Remember to multiply the probability of 10 or more successes   
# by 2 for the two-tailed test result.  
  
2 \* sum(probx[xsuccesses >= 10])

## [1] 1.987976e-06

# For a faster result, try R's built-in binomial test.   
# The resulting P-value is slightly different from our calculation. # The output of binom.test includes a confidence interval for the   
# proportion using the Clopper-Pearson method, which is more   
# conservative than the Agresti-Coull method.  
  
binom.test(10, n = 25, p = 0.061)

##   
## Exact binomial test  
##   
## data: 10 and 25  
## number of successes = 10, number of trials = 25, p-value = 9.94e-07  
## alternative hypothesis: true probability of success is not equal to 0.061  
## 95 percent confidence interval:  
## 0.2112548 0.6133465  
## sample estimates:  
## probability of success   
## 0.4

# Agresti-Coull 95% confidence interval for the proportion   
# using the binom package.  
  
library(binom)  
binom.confint(30, n = 87, method = "ac")

## method x n mean lower upper  
## 1 agresti-coull 30 87 0.3448276 0.2532164 0.4495625

## test

We perform a goodness-of-fit test. We use a Poisson probability model and fit it to frequency data on the number of marine invertebrate extinctions per time block in the fossil record. We first read and inspect the data. Each row is a time block, with the observed number of extinctions listed.

The first step is to create a frequency table for the number of time blocks in each number-of-extinctions category. The command table does what’s needed, but note that some extinction categories are not represented (e.g., 0, 12 and 13 extinctions).

extinctTable <- table(extinctData$numberOfExtinctions)  
data.frame(Frequency = addmargins(extinctTable))

## Frequency.Var1 Frequency.Freq  
## 1 1 13  
## 2 2 15  
## 3 3 16  
## 4 4 7  
## 5 5 10  
## 6 6 4  
## 7 7 2  
## 8 8 1  
## 9 9 2  
## 10 10 1  
## 11 11 1  
## 12 14 1  
## 13 16 2  
## 14 20 1  
## 15 Sum 76

To remedy the problem of missing categories, we transform the original variable into a factor (that is, a categorical variable) with all counts between 0 and 20 as levels (note that 20 is not the maximum possible number of extinctions, but it is a convenient cutoff for this table).

extinctData = mutate(extinctData,nExtinctFactor = factor(numberOfExtinctions, levels = c(0:20)))  
   
extinctTable2 <- table(extinctData$nExtinctFactor)

Estimate the mean number of extinctions per time block from the data. The estimate is needed for the goodness-of-fit test.

meanExtinctions <- mean(extinctData$numberOfExtinctions)  
meanExtinctions

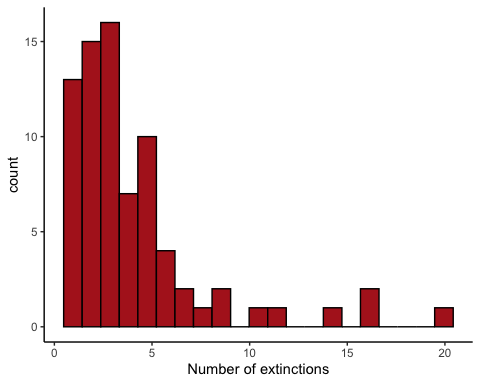
## [1] 4.210526

Calculate expected frequencies under the Poisson distribution using the estimated mean. (For now, continue to use 20 extinctions as the cutoff, but don’t forget that the Poisson distribution includes the number-of-extinctions categories 21, 22, 23, and so on.)

expectedProportion <- dpois(0:20, lambda = meanExtinctions)  
expectedFrequency <- expectedProportion \* 76

Show the frequency distribution in a histogram.

ggplot(extinctData) +  
 geom\_histogram(mapping= aes(x =numberOfExtinctions),fill = "firebrick",color="black",bins=21)+   
 labs (x="Number of extinctions")



Make a table of observed and expected frequencies, saving as a data frame.

extinctFreq <- data.frame(nExtinct = 0:20, obsFreq = as.vector(extinctTable2),   
 expFreq = expectedFrequency)  
extinctFreq

## nExtinct obsFreq expFreq  
## 1 0 0 1.127730e+00  
## 2 1 13 4.748338e+00  
## 3 2 15 9.996501e+00  
## 4 3 16 1.403018e+01  
## 5 4 7 1.476861e+01  
## 6 5 10 1.243672e+01  
## 7 6 4 8.727524e+00  
## 8 7 2 5.249639e+00  
## 9 8 1 2.762968e+00  
## 10 9 2 1.292616e+00  
## 11 10 1 5.442596e-01  
## 12 11 1 2.083290e-01  
## 13 12 0 7.309790e-02  
## 14 13 0 2.367543e-02  
## 15 14 1 7.120431e-03  
## 16 15 0 1.998718e-03  
## 17 16 2 5.259783e-04  
## 18 17 0 1.302733e-04  
## 19 18 0 3.047328e-05  
## 20 19 0 6.753081e-06  
## 21 20 1 1.421701e-06

The low expected frequencies will violate the assumptions of the test, so we will need to group categories. Create a new variable that groups the extinctions into fewer categories (again, there are many ways to do this, here I use the function *cut*).

extinctFreq$groups <- cut(extinctFreq$nExtinct,   
 breaks = c(0, 2:8, 21), right = FALSE,  
 labels = c("0 or 1","2","3","4","5","6","7","8 or more"))  
extinctFreq

## nExtinct obsFreq expFreq groups  
## 1 0 0 1.127730e+00 0 or 1  
## 2 1 13 4.748338e+00 0 or 1  
## 3 2 15 9.996501e+00 2  
## 4 3 16 1.403018e+01 3  
## 5 4 7 1.476861e+01 4  
## 6 5 10 1.243672e+01 5  
## 7 6 4 8.727524e+00 6  
## 8 7 2 5.249639e+00 7  
## 9 8 1 2.762968e+00 8 or more  
## 10 9 2 1.292616e+00 8 or more  
## 11 10 1 5.442596e-01 8 or more  
## 12 11 1 2.083290e-01 8 or more  
## 13 12 0 7.309790e-02 8 or more  
## 14 13 0 2.367543e-02 8 or more  
## 15 14 1 7.120431e-03 8 or more  
## 16 15 0 1.998718e-03 8 or more  
## 17 16 2 5.259783e-04 8 or more  
## 18 17 0 1.302733e-04 8 or more  
## 19 18 0 3.047328e-05 8 or more  
## 20 19 0 6.753081e-06 8 or more  
## 21 20 1 1.421701e-06 8 or more

Then sum up the observed and expected frequencies within the new categories.

obsFreqGroup <- tapply(extinctFreq$obsFreq, extinctFreq$groups, sum)  
expFreqGroup <- tapply(extinctFreq$expFreq, extinctFreq$groups, sum)  
ObsVsExp = data.frame(obs = obsFreqGroup, exp = expFreqGroup,group= c("0 or 1","2","3","4","5","6","7","8 or more"))  
ObsVsExp

## obs exp group  
## 0 or 1 13 5.876068 0 or 1  
## 2 15 9.996501 2  
## 3 16 14.030177 3  
## 4 7 14.768608 4  
## 5 10 12.436722 5  
## 6 4 8.727524 6  
## 7 2 5.249639 7  
## 8 or more 9 4.914760 8 or more

The *expected* frequency for the last category, “8 or more”, doesn’t yet include the expected frequencies for the categories 21, 22, 23, and so on (remeber that we used a cut-off of 20 before!). However, the expected frequencies must sum to 76. In the following, we recalculate the expected frequency for the last group, expFreqGroup[length(expFreqGroup)], as 76 minus the sum of the expected frequencies for all the other groups.

expFreqGroup[length(expFreqGroup)] = 76 - sum(expFreqGroup[1:(length(expFreqGroup)-1)])  
data.frame(obs = obsFreqGroup, exp = expFreqGroup)

## obs exp  
## 0 or 1 13 5.876068  
## 2 15 9.996501  
## 3 16 14.030177  
## 4 7 14.768608  
## 5 10 12.436722  
## 6 4 8.727524  
## 7 2 5.249639  
## 8 or more 9 4.914761

Finally, we are ready to carry out the goodness-of-fit test. R gives us a warning here because one of the expected frequencies is less than 5. However, we have been careful to meet the assumptions of the test, so let’s persevere. Once again, R doesn’t know that we have estimated a parameter from the data (the mean), so it won’t use the correct degrees of freedom when calculating the P-value. As before, we need to grab the value calculated by chisq.test and recalculate using the correct degrees of freedom. Since the number of categories is now 8, the correct degrees of freedom is 8 - 1 - 1 = 6.

saveChiTest <- chisq.test(obsFreqGroup, p = expFreqGroup/76)

## Warning in chisq.test(obsFreqGroup, p = expFreqGroup/76): Chi-squared  
## approximation may be incorrect

saveChiTest

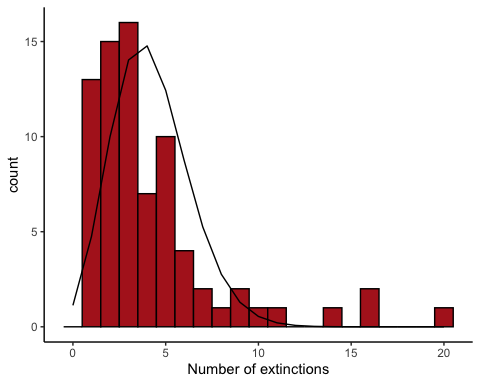
##   
## Chi-squared test for given probabilities  
##   
## data: obsFreqGroup  
## X-squared = 23.95, df = 7, p-value = 0.001163

# Wrong degrees of freedom, so wrong P-value!  
  
pValue <- 1 - pchisq(saveChiTest$statistic, df = 6)  
  
# correct P-value!  
pValue

## X-squared   
## 0.0005334919

Finally, we can compare the expected and observed data in a histogramm. One can clearly see that the fit is not very good.

df = data.frame(x = 0:20,y = 76\*dpois(0:20,meanExtinctions))  
  
ggplot(extinctData) +  
 geom\_histogram(mapping= aes(x =numberOfExtinctions),fill = "firebrick",color="black",bins=21)+   
 labs (x="Number of extinctions") +   
 geom\_line(data = df, mapping = aes(x = x,y=y))



## t-test

### One-sample t-test

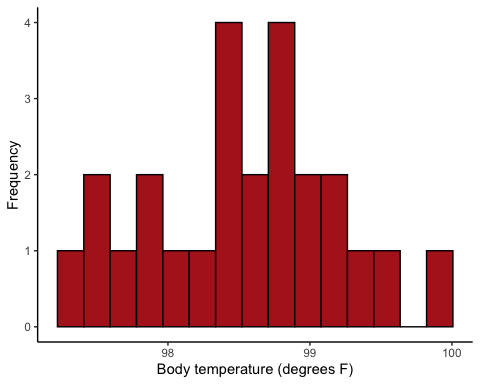
We use a one-sample t-test to compare body temperature in a random sample of people with the “expected” temperature 98.6 F.

Read and inspect the data:

heat <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter11/chap11e3Temperature.csv"))  
head(heat)

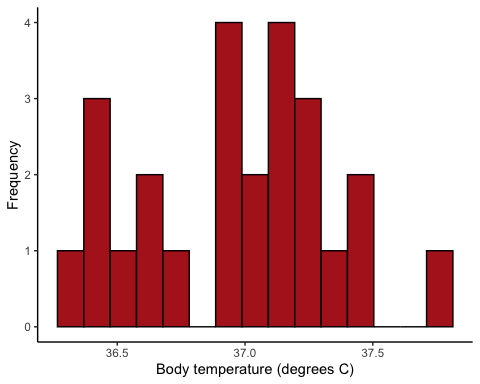
## individual temperature  
## 1 1 98.4  
## 2 2 98.6  
## 3 3 97.8  
## 4 4 98.8  
## 5 5 97.9  
## 6 6 99.0

ggplot(data = heat) +   
 geom\_histogram(mapping = aes(x = temperature),fill = "firebrick",color="black",bins=15) +   
 labs(x = "Body temperature (degrees F)",y = "Frequency", main = "")



A one-sample t-test can be calculate using the function t.test. The mu argument gives the value stated in the null hypothesis. Let us first transform the data into celsius using the formula to turn:

mu0 = (98.6-32)\*5/9  
  
heat = mutate(heat,temperatureC = (temperature-32) \* 5/9)  
  
ggplot(data = heat) +   
 geom\_histogram(mapping = aes(x = temperatureC),fill = "firebrick",color="black",bins=15) +   
 labs(x = "Body temperature (degrees C)",y = "Frequency", main = "")



t.test(heat$temperatureC, mu = mu0)

##   
## One Sample t-test  
##   
## data: heat$temperatureC  
## t = -0.56065, df = 24, p-value = 0.5802  
## alternative hypothesis: true mean is not equal to 37  
## 95 percent confidence interval:  
## 36.80235 37.11321  
## sample estimates:  
## mean of x   
## 36.95778

### Two-sample t-test

We compare horn length of live and dead (spiked) horned lizards.

lizard <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter12/chap12e3HornedLizards.csv"))  
head(lizard)

## squamosalHornLength Survival  
## 1 25.2 living  
## 2 26.9 living  
## 3 26.6 living  
## 4 25.6 living  
## 5 25.7 living  
## 6 25.9 living

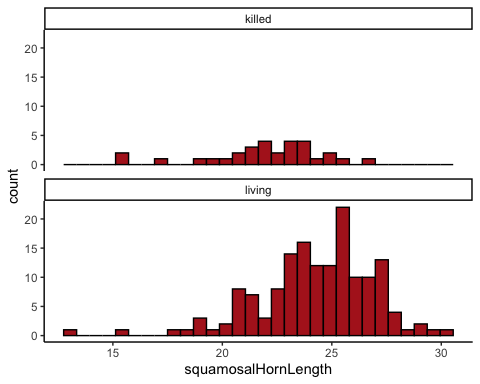
Note that there is one missing value for the variable “squamosalHornLength”. Everything is easier if we eliminate the row with missing data.

lizard2 <- na.omit(lizard)  
head(lizard2)

## squamosalHornLength Survival  
## 1 25.2 living  
## 2 26.9 living  
## 3 26.6 living  
## 4 25.6 living  
## 5 25.7 living  
## 6 25.9 living

ggplot(data = lizard2) +  
 geom\_histogram(mapping = aes(x = squamosalHornLength ),fill = "firebrick",color="black") +   
 facet\_wrap( ~Survival,nrow=2) +   
 theme\_classic()

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



A two-sample t-test of the difference between two means can be carried out with t.test by using a formula, asking if squamosalHornLength is predicted by Survival, and specifying that the variables are in the data frame lizard.

t.test(squamosalHornLength ~ Survival, data = lizard)

##   
## Welch Two Sample t-test  
##   
## data: squamosalHornLength by Survival  
## t = -4.2634, df = 40.372, p-value = 0.0001178  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -3.381912 -1.207092  
## sample estimates:  
## mean in group killed mean in group living   
## 21.98667 24.28117

The same reuslt could be achieved with

t.test(lizard$squamosalHornLength[lizard$Survival=="killed"], lizard$squamosalHornLength[lizard$Survival=="living"])

##   
## Welch Two Sample t-test  
##   
## data: lizard$squamosalHornLength[lizard$Survival == "killed"] and lizard$squamosalHornLength[lizard$Survival == "living"]  
## t = -4.2634, df = 40.372, p-value = 0.0001178  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -3.381912 -1.207092  
## sample estimates:  
## mean of x mean of y   
## 21.98667 24.28117

The output of t.test includes the 95% confidence interval for the difference between means. Add $confint after calling the function to get R to report only the confidence interval. The formula in the following command tells R to compare squamosalHornLength between the two groups indicated by Survival.

t.test(squamosalHornLength ~ Survival, data = lizard)$conf.int

## [1] -3.381912 -1.207092  
## attr(,"conf.level")  
## [1] 0.95

\*Note: R has used the Welch two sample t-test here. If we want to force R to use the standrad t.test we set a parameter specifiying this:

t.test(squamosalHornLength ~ Survival, data = lizard,var.equal = TRUE)

##   
## Two Sample t-test  
##   
## data: squamosalHornLength by Survival  
## t = -4.3494, df = 182, p-value = 2.27e-05  
## alternative hypothesis: true difference in means is not equal to 0  
## 95 percent confidence interval:  
## -3.335402 -1.253602  
## sample estimates:  
## mean in group killed mean in group living   
## 21.98667 24.28117

# Deviations from Normailty

## QQ Plots and Transformations

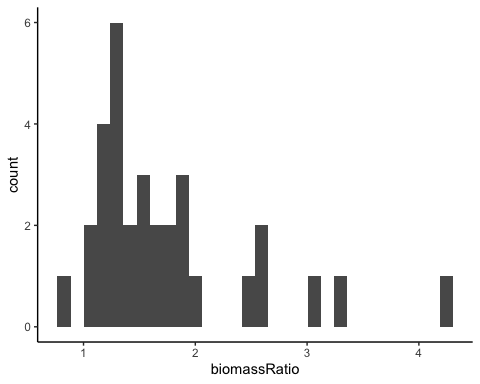
We investigate the ratio of biomass between marine reserves and non-reserve control areas.

marine <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter13/chap13e1MarineReserve.csv"))  
head(marine)

## biomassRatio  
## 1 1.34  
## 2 1.96  
## 3 2.49  
## 4 1.27  
## 5 1.19  
## 6 1.15

ggplot(data = marine) +  
 geom\_histogram(mapping = aes(x = biomassRatio))

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.

 Normal quantile plot of biomass ratio data.

library(car)

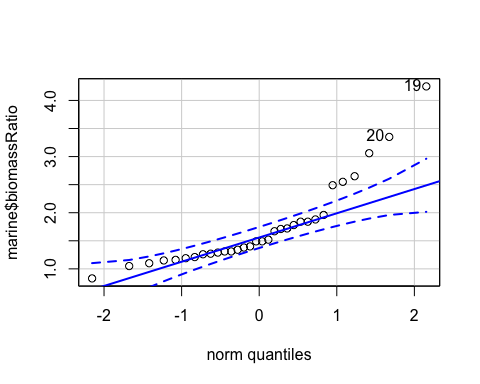
## Loading required package: carData

##   
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':  
##   
## recode

## The following object is masked from 'package:purrr':  
##   
## some

qqPlot(marine$biomassRatio, distribution = "norm")



## [1] 19 20

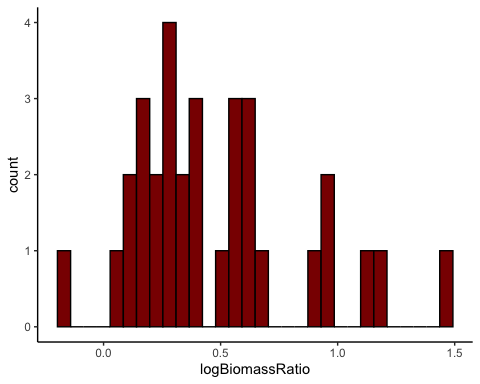
The function log() takes the natural logarithm of all the elements of a vector or variable. The following command puts the results into a new variable in the same data frame, marine.

marine = mutate(marine,logBiomassRatio = log(biomassRatio))

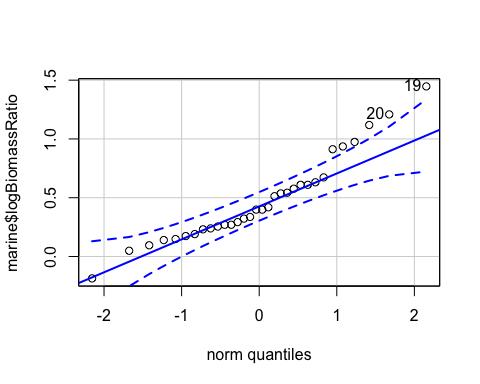
Histogram and QQ-Plot of the log-transformed marine biomass ratio.

ggplot(data = marine) +  
 geom\_histogram(mapping = aes(x = logBiomassRatio),col="black",fill="darkred")

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



qqPlot(marine$logBiomassRatio, distribution = "norm")



## [1] 19 20

95% confidence interval of the mean using the log-transformed data.

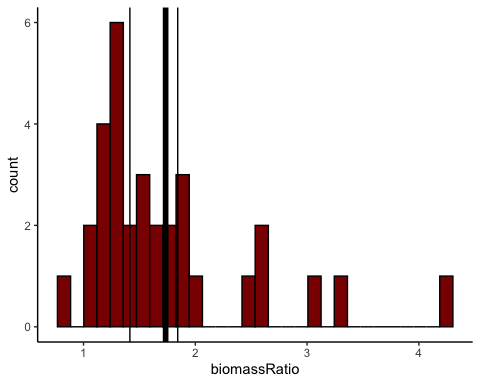
t.test(marine$logBiomassRatio)$conf.int

## [1] 0.3470180 0.6112365  
## attr(,"conf.level")  
## [1] 0.95

Back-transform lower and upper limits of confidence interval (exp is the inverse of the log function).

conf.int = exp(t.test(marine$logBiomassRatio)$conf.int )  
meanBiomassRatio = mean(marine$biomassRatio)  
  
ggplot(data = marine) +   
 geom\_histogram(mapping= aes(x = biomassRatio),col="black",fill="darkred") +   
 geom\_vline(xintercept = conf.int)+   
 geom\_vline(xintercept = meanBiomassRatio,lwd=2)

## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



## Non-Parametric Alternatives

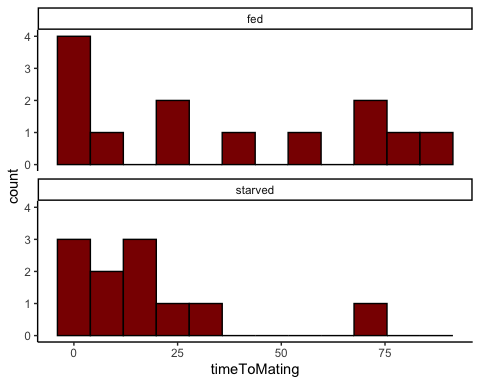
We use the Wilcoxon rank-sum test (equivalent to the Mann-Whitney U-test) comparing times to mating (in hours) of starved and fed female sagebrush crickets. We also apply the permutation test to the same data.

Read and inspect data:

cannibalism <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter13/chap13e5SagebrushCrickets.csv"))  
head(cannibalism)

## feedingStatus timeToMating  
## 1 starved 1.9  
## 2 starved 2.1  
## 3 starved 3.8  
## 4 starved 9.0  
## 5 starved 9.6  
## 6 starved 13.0

ggplot(data = cannibalism) +  
 geom\_histogram(mapping = aes(x = timeToMating ),bins=12,color="black",fill="darkred") +   
 facet\_wrap( ~feedingStatus,nrow=2) +   
 theme\_classic()



We perfrom the Wilcoxon rank-sum test (equivalent to Mann-Whitney U-test)

wilcox.test(timeToMating ~ feedingStatus, data = cannibalism)

##   
## Wilcoxon rank sum test  
##   
## data: timeToMating by feedingStatus  
## W = 88, p-value = 0.3607  
## alternative hypothesis: true location shift is not equal to 0

## Permutation Test

We use a permutation test of the difference between mean time to mating of starved and fed crickets.We begin by calculating the observed difference between means (starved minus fed). The difference is -18.25734 in this data set.

# tapply is another way to calcualte the means of time to mating  
# for each feeding status seperatly  
  
cricketMeans <- tapply(cannibalism$timeToMating, cannibalism$feedingStatus, mean)  
  
cricketMeans

## fed starved   
## 35.98462 17.72727

diffMeans <- cricketMeans[2] - cricketMeans[1]  
diffMeans

## starved   
## -18.25734

We choose to perform 1000 permutations and set a seed to make the analysis reproducible:

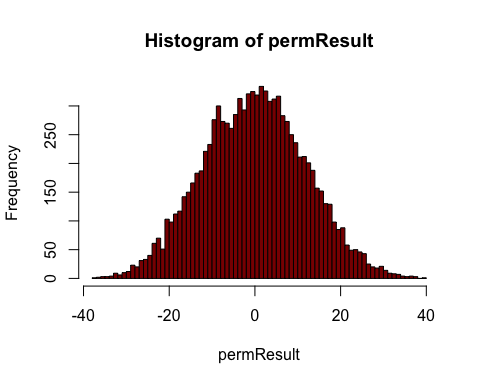
nPerm <- 10000  
set.seed(2048)

The following code implements aloop to permute the data many times (determined by nperm, investigate the function sample to see what it does). In the loop, i is just a counter that goes from 1 to nPerm by 1; each permuted difference is saved in the permResult.

permResult <- vector() # initializes  
for(i in 1:nPerm){  
 # step 1: permute the times to mating  
 permSample <- sample(cannibalism$timeToMating, replace = FALSE)  
 # step 2: calculate difference betweeen means  
 permMeans <- tapply(permSample, cannibalism$feedingStatus, mean)  
 permResult[i] <- permMeans[2] - permMeans[1]  
}

Plot null distribution based on the permuted differences.

hist(permResult, right = FALSE, breaks = 100,col="darkred")



Use the null distribution to calculate an approximate P-value. This is the twice the proportion of the permuted means that fall below the observed difference in means, diffMeans (-18.25734 in this example). The following code calculates the number of permuted means falling below diffMeans.

sum(as.numeric(permResult <= diffMeans))

## [1] 657

These commands obtain the fraction of permuted means falling below diffMeans.

sum(as.numeric(permResult <= diffMeans)) / nPerm

## [1] 0.0657

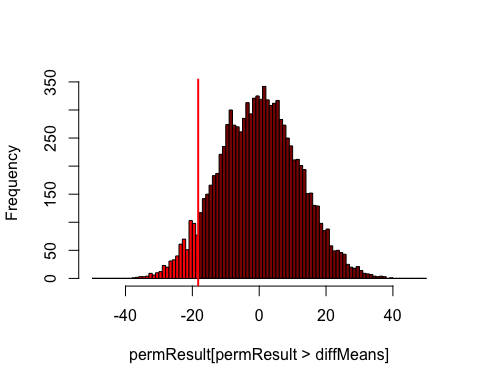
Finally, multiply by 2 to get the P-value for a two-sided test.

2 \* ( sum(as.numeric(permResult <= diffMeans)) / nPerm )

## [1] 0.1314

We can illustrate the permutaed differences and add the observed difference on to show the outcome of our test: the birght red colored bars indicate the proportion of permutation results that yielded a more exterme results as the on observed.

hist(permResult[permResult>diffMeans],breaks=seq(-50,50,by=1),col="darkred",main="")  
hist(permResult[permResult<=diffMeans],breaks=seq(-50,50,by=1),add=T,col="red")  
abline(v = diffMeans,col="red",lwd=2)



# ANOVA

## One way ANOVA

We compare the phase shift in the circadian rhythm of melatonin production in participants given alternative light treatments.

Read and inspect the data.

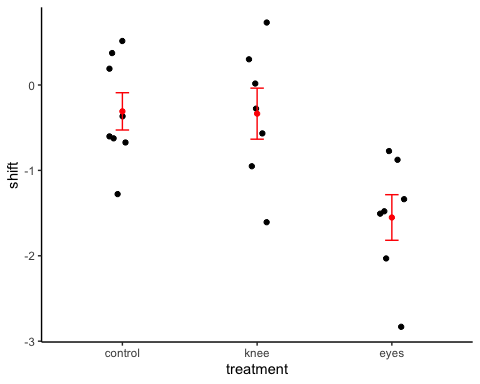
circadian <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter15/chap15e1KneesWhoSayNight.csv"))

Set the preferred ordering of groups in tables and graphs.

circadian$treatment <- factor(circadian$treatment,   
 levels = c("control", "knee", "eyes"))   
  
  
  
meanShift <- tapply(circadian$shift, circadian$treatment, mean)  
sdevShift <- tapply(circadian$shift, circadian$treatment, sd)  
n <- tapply(circadian$shift, circadian$treatment, length)  
mean\_SD =data.frame(mean = meanShift, std.dev = sdevShift, n = n,treatment = c("control", "knee", "eyes"))

We plot the data showing the raw data points as well as the mean (including error bars shwoing one standard error in each direction).

ggplot(data = circadian) +  
 geom\_jitter(mapping = aes(x = treatment,y=shift),width=0.1) +   
 geom\_errorbar(data = mean\_SD,mapping = aes(x = treatment,ymin=meanShift-sdevShift/sqrt(n), ymax=meanShift+sdevShift/sqrt(n)), width=.1,color="red") +   
 geom\_point(data = mean\_SD,mapping = aes(x = treatment,y=meanShift),color="red")



The first step involves fitting the ANOVA model to the data using lm (“lm” stands for “linear model”, of which ANOVA is one type). Then we use the command anova to assemble the ANOVA table.

circadianAnova <- lm(shift ~ treatment, data = circadian)  
anova(circadianAnova)

## Analysis of Variance Table  
##   
## Response: shift  
## Df Sum Sq Mean Sq F value Pr(>F)   
## treatment 2 7.2245 3.6122 7.2894 0.004472 \*\*  
## Residuals 19 9.4153 0.4955   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

R2 indicating the fraction of variation in the response variable “explained” by treatment. This is again done in two steps. The first step calculates a bunch of useful quantities from the ANOVA model object previously created with a lm command. The second step shows the R2 value.

circadianAnovaSummary <- summary(circadianAnova)  
circadianAnovaSummary$r.squared

## [1] 0.4341684

## Two way ANOVA

Analyze data from a factorial experiment investigating the effects of herbivore presence, height above low tide, and the interaction between these factors, on abundance of a red intertidal alga using two-factor ANOVA.

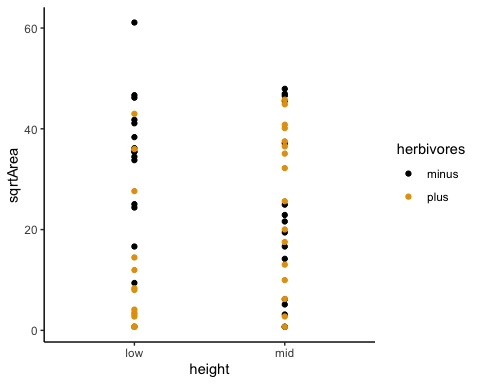
Read and inspect the data.

algae <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter18/chap18e3IntertidalAlgae.csv"))  
head(algae)

## height herbivores sqrtArea  
## 1 low minus 9.405573  
## 2 low minus 34.467736  
## 3 low minus 46.673485  
## 4 low minus 16.642139  
## 5 low minus 24.377498  
## 6 low minus 38.350604

We first take a look at the data:

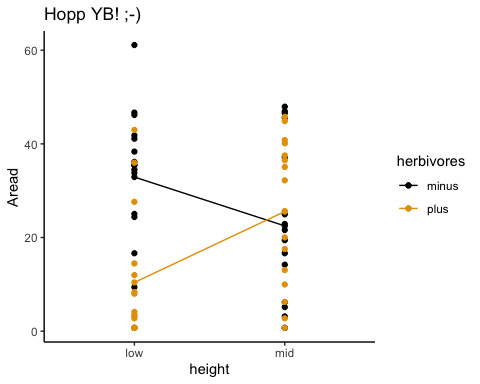
ggplot(data = algae, aes(x = height, color = herbivores, group = herbivores, y = sqrtArea)) +  
 geom\_point()



This is a mess so we next add some summary statistics of the data: the mean Area for each combination of explanatory variables.

ggplot(data = algae) +  
 aes(x = height, color = herbivores, group = herbivores, y = sqrtArea) +  
 geom\_point()+  
 stat\_summary(fun.y = mean, geom = "point") +  
 stat\_summary(fun.y = mean, geom = "line")+   
 labs(title="Hopp YB! ;-)",  
 x ="height", y = "Aread")

## Warning: `fun.y` is deprecated. Use `fun` instead.  
  
## Warning: `fun.y` is deprecated. Use `fun` instead.



It looks like there is some interaction between *height* and *herbivores*, but there is also quite some noise so it is difficult to tell.

Let us first fit a null model having both main effects but no interaction term. We can do this again with *lm* (*note:* conceptually a t-test, an ANOVA and linear regression are all pretty much the same thing and hence the function lm can be used in all these cases):

algaeNoInteractModel <- lm(sqrtArea ~ height + herbivores, data = algae)  
  
summary(algaeNoInteractModel)

##   
## Call:  
## lm(formula = sqrtArea ~ height + herbivores, data = algae)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -28.171 -13.748 -2.235 15.433 34.576   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 26.520 3.602 7.362 5.53e-10 \*\*\*  
## heightmid 2.358 4.160 0.567 0.5729   
## herbivoresplus -9.722 4.160 -2.337 0.0227 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 16.64 on 61 degrees of freedom  
## Multiple R-squared: 0.0866, Adjusted R-squared: 0.05665   
## F-statistic: 2.892 on 2 and 61 DF, p-value: 0.06311

Now fit the full model, with interaction term included.

algaeFullModel <- lm(sqrtArea ~ height \* herbivores, data = algae)

Finally we cretea na ANOVA table that compares the two models:

anova(algaeNoInteractModel, algaeFullModel)

## Analysis of Variance Table  
##   
## Model 1: sqrtArea ~ height + herbivores  
## Model 2: sqrtArea ~ height \* herbivores  
## Res.Df RSS Df Sum of Sq F Pr(>F)   
## 1 61 16888   
## 2 60 14270 1 2617 11.003 0.001549 \*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

# Linear Regression

We test the null hypothesis of zero regression slope. The data are from an experiment investigating the effect of plant species diversity on the stability of plant biomass production.

Read and inspect the data.

prairie <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter17/chap17e3PlantDiversityAndStability.csv"))  
head(prairie)

## nSpecies biomassStability  
## 1 1 7.47  
## 2 1 6.74  
## 3 1 6.61  
## 4 1 6.40  
## 5 1 5.67  
## 6 1 5.26

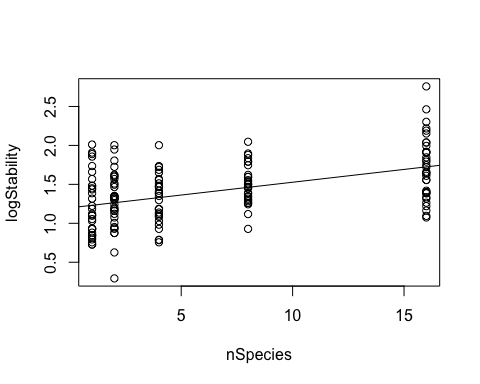
We perfrom a Log-transform on the varibale stability and include the new variable in the data frame. Inspect the data frame once again to make sure the function worked as intended.

prairie = mutate(prairie,logStability = log(biomassStability))  
head(prairie)

## nSpecies biomassStability logStability  
## 1 1 7.47 2.010895  
## 2 1 6.74 1.908060  
## 3 1 6.61 1.888584  
## 4 1 6.40 1.856298  
## 5 1 5.67 1.735189  
## 6 1 5.26 1.660131

Scatter plot with regression line in base R:

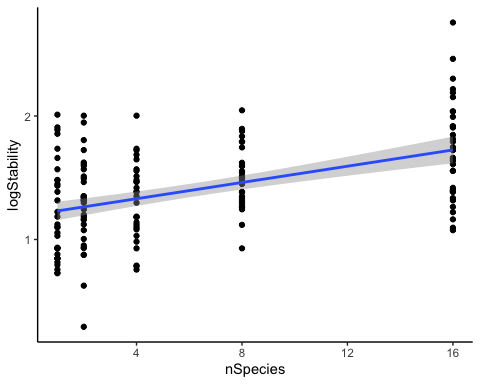
plot(logStability ~ nSpecies, data = prairie)   
prairieRegression <- lm(logStability ~ nSpecies, data = prairie)  
abline(prairieRegression)



Scatter plot with regression line in ggplot:

ggplot(data = prairie,aes(y=logStability,x=nSpecies)) +  
 geom\_point()+  
 geom\_smooth(method = "lm")

## `geom\_smooth()` using formula 'y ~ x'



prairieRegression <- lm(logStability ~ nSpecies, data = prairie)  
summary(prairieRegression)

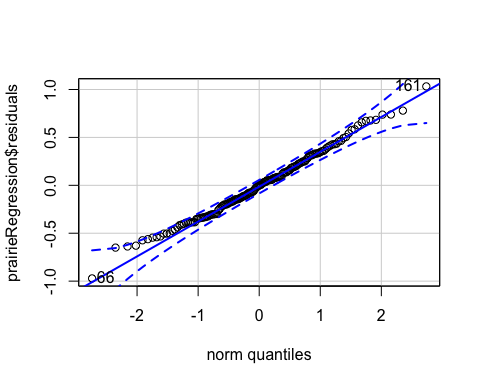
##   
## Call:  
## lm(formula = logStability ~ nSpecies, data = prairie)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.97148 -0.25984 -0.00234 0.23100 1.03237   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.198294 0.041298 29.016 < 2e-16 \*\*\*  
## nSpecies 0.032926 0.004884 6.742 2.73e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3484 on 159 degrees of freedom  
## Multiple R-squared: 0.2223, Adjusted R-squared: 0.2174   
## F-statistic: 45.45 on 1 and 159 DF, p-value: 2.733e-10

confint(prairieRegression)

## 2.5 % 97.5 %  
## (Intercept) 1.11673087 1.27985782  
## nSpecies 0.02328063 0.04257117

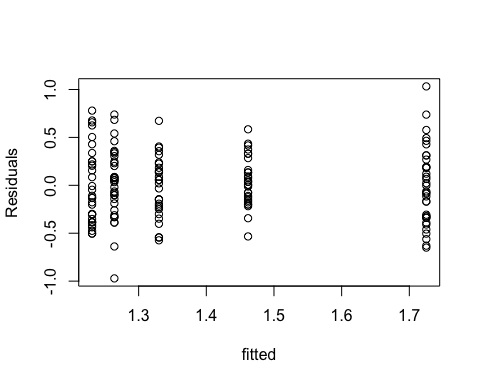
Finally, we check our modelling assumptions with a QQ plot of the residuals as well as a Tukey-Anscombe plot:

qqPlot(prairieRegression$residuals,dist="norm")



## [1] 161 66

plot(prairieRegression$fitted.values,prairieRegression$residuals,xlab="fitted",ylab="Residuals")



# Additional notes to the course slides

## Law of total probability

Assume that is a set of pairwise disjoint events whose union is the entire sample space (that is, every possible event that can happen is covered by exactly one of the s).

As an example: consider rolling a six-sided die. Then the event is the event that you roll the number . All events are disjoint and together they cover everything that can happen.

For any event we can then write

or, equivalently,

This figure is helpful for understanding what is going on here:

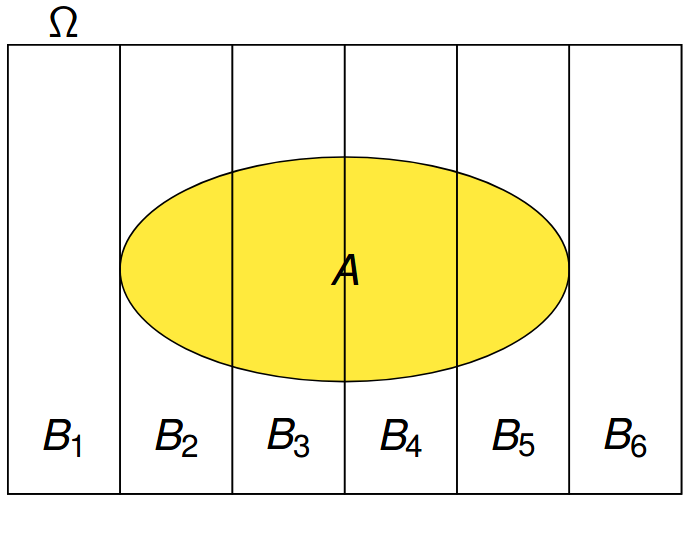


Illustration of the law of total probability

Example:

We roll two dice. We want to know the probaility that the first roll has a smaller outcome than the second roll.

We call the events the event that the frist roll shows dots. So is the event that you roll a 2 with the first roll. (Think about why this satisfies the conditons for the events stated in the law of total probaility.)

Let us clall the outcome of the first roll and the outcome of the second roll: is then . The event is the event that . We use the formula

If you roll a with the first die, there are 6- possible outcomes for the second roll that are larger than . This means:

and furthermore

for each .

Then

Let us test this with a small R simulation:

# We do 10000 die rolls for each die:  
die.1 = sample.int(6, size = 10000,replace=T)  
die.2 = sample.int(6, size = 10000,replace=T)  
  
# calcualte the proportion of times die 2 is larger than die 1  
sum(die.2>die.1)/10000

## [1] 0.4129

# Exercises

## Week 1 (24.02.2021)

#### Exercise 1

The peppered moth (Biston betularia) occurs in two types: peppered (speckled black and white) and melanic (black). A researcher wished to measure the proportion of melanic individuals in the peppered moth population in England, to examine how this proportion changed from year to year in the past. To accomplish this, she photographed all the peppered moth specimens available in museums and large private collections and grouped them by the year in which they had been collected. Based on this sample, she calculated the proportion of melanic individuals in every year. The people who collected the specimens, she knew, would prefer to collect whichever type was rarest in any given year, since those would be the most valuable.

1. Can the specimens from any given year be considered a random sample from the moth population?
2. If not a random sample, what type of sample is it?
3. What type of error might be introduced by the sampling method when estimating the proportion of melanic moths?

#### Exercise 2

Which of the following numerical variables are continuous? Which are discrete? (Note: Discrete variables are those that have a finite set of predefined values, whereas continuous variables can take any value within a given interval.)

1. Number of injuries sustained in a fall
2. Fraction of birds in a large sample infected with avian flu virus
3. Number of crimes committed by a randomly sampled individual
4. Logarithm of body mass

#### Exercise 3

For each of the following studies, say which is the explanatory variable and which is the response variable. Also, say whether the study is observational or experimental.

1. Forestry researchers wanted to compare the growth rates of trees growing at high altitude to that of trees growing at low altitude. They measured growth rates using the space between tree rings in a set of trees harvested from a natural forest.
2. Researchers randomly assign diabetes patients to two groups. In the first group, the patients receive a new drug, tasploglutide, whereas patients in the other group receive standard treatment without the new drug. The researchers compared the rate of insulin release in the two groups.
3. Psychologists tested whether the frequency of illegal drug use differs between people suffering from schizophrenia and those not having the disease. They measured drug use in a group of schizophrenia patients and compared it with that in a similar sized group of randomly chosen people.
4. Spinner Hansen et al. (2008) studied a species of spider whose females often eat males that are trying to mate with them. The researchers removed a leg from each male spider in one group (to make them weaker and more vulnerable to being eaten) and left the males in another group undamaged. They studied whether survival of males in the two groups differed during courtship.
5. Bowen et al. (2012) studied the effects of advanced communication therapy for patients whose communication skills had been affected by previous strokes. They randomly assigned two therapies to stroke patients. One group received advanced communication therapy and the other received only social visits without formal therapy. Both groups otherwise received normal, best-practice care. After six months, the communication ability (as measured by a standardized quantitative test score) was measured on all patients.

#### Exercise 4

During World War II, the British Royal Air Force estimated the density of bullet holes on different sections of planes returning to base from aerial sorties. Their goal was to use this information to determine which plane sections most needed additional protective shields. (It was not possible to reinforce the whole plane, because it would weigh too much.) They found that the density of holes was highest on the wings and lowest on the engines and near the cockpit, where the pilot sits (their initial conclusion, that therefore the wings should be reinforced, was later shown to be mistaken). What is the main problem with the sample: bias or large sampling error? What part of the plane should have been reinforced?

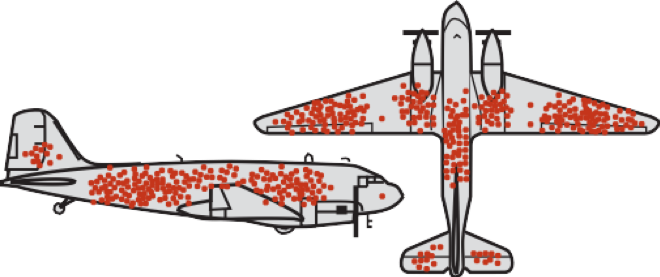


Figure for exercise 4: Airplane bullet holes

#### Exercise 5

An important quantity in conservation biology is the number of plant and animal species inhabiting a given area. To survey the community of small mammals inhabiting Kruger National Park in South Africa, a large series of live traps were placed randomly throughout the park for one week in the main dry season of 2004. Traps were set each evening and checked the following morning. Individuals caught were identified, tagged (so that new captures could be distinguished from recaptures), and released. At the end of the survey, the total number of small mammal species in the park was estimated by the total number of species captured in the survey.

1. What is the parameter being estimated in the survey?
2. Is the sample of individuals captured in the traps likely to be a random sample? Why or why not? In your answer, address the two criteria that define a sample as random.
3. Is the number of species in the sample likely to be an unbiased estimate of the total number of small mammal species in the park?

### Solutions

#### Exercise 1

1. No. Collectors did not choose randomly, but prefered rarer types over more common ones.
2. It is a sample of convenience.
3. Thus, there is bias in every year’s sample because rare types are over represerented. Further this bias might change across years as the frequencies of the two morphs have changed over time.

#### Exercise 2

1. Discrete.
2. Technically it is a discrete variable (because fractions can be enumerated / are retricted to fintely many values in a fintie sample) but it makes sense to treat it as a continous variable if the sample is very large.
3. Discrete. There are no half crimes.
4. Continous. Body mass is continous and hence the log as well.

#### Exercise 3

1. E(xplanatory): Altitude (categorical: high vs low) R(esponse): Growth rate S(tudy type): Observational
2. E: Treatment (standard vs. tasploglutide) R: Rate of insulin release S: Experimental
3. E: Health status (schizophrenia vs. healthy) R: Frequency of illegal drug use S: Observational
4. E:Number of legs R: Survival propability S: Experimental
5. E: Treatment (advanced communication therapy vs. social visits without formal therapy) R: Communication ability S: Experimental

#### Exercise 4

The main problem is a strong bias in the sample. To see this, consider the sample of planes that were used in this study. Only planes that were *not* hit in critical areas were available to estimate the distribution of bullet holes. Planes that were hit in a critical area, i.e., one that leads to a crash, were not available because they did not return to base. With this knowledge, it becomes clear that it would have been better to reinforce the areas were no, or very little, bullet holes were found, namely the cockpit and engine.

#### Exercise 5

1. The population parameter being estimated is all the small mammals of Kruger National Park.
2. No, the sample is not likely to be random. In a random sample, every individual has the same chance of being selected. But some small mammals might be easier to trap than others (for example, trapping only at night might miss all the mammals active only in the day time). In a random sample individuals are selected independently. Multiple animals caught in the same trap might not be independent if they are related or live near one another (this is harder to judge).
3. The number of species in the sample might underestimate the number in the Park if sampling was not random (e.g., if daytime mammals were missed), or if rare species happened to avoid capture. Even if the sample is perfectly random, the estimator will underestimated the true number in most cases. You can sample fewer species just by chance, but not more species as there actually are. Thus - on average ??? you will underestimate the true number.

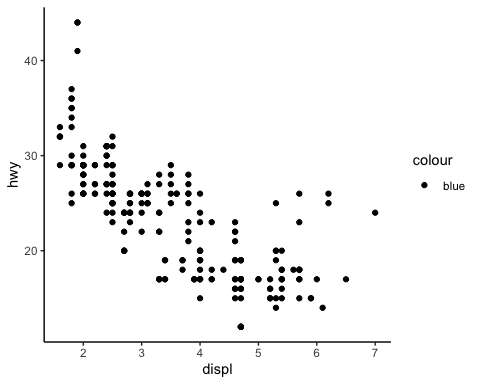
## Week 2 (03.03.2021)

#### Exercise 6

We use the MPG data set introduced in the lecture. Answer the following questions. You can use base R or ggplot if not otherwise specified.

1. Run ggplot(data = mpg). What do you see?
2. How many rows are in mpg? How many columns?
3. What does the drv variable describe? Read the help for ?mpg to find out.
4. Make a scatterplot of hwy vs cyl.
5. What happens if you make a scatterplot of class vs drv? Why is the plot not useful?
6. What is wrong with this code? Why are the points not blue?

ggplot(data = mpg) +   
  
 geom\_point(mapping = aes(x = displ, y = hwy, color = "blue"))



1. Map a continuous variable to color, size, and shape. How do these aesthetics behave differently for categorical vs. continuous variables?
2. Fix the following code:

ggplot(data = mpg)   
  
 + geom\_point(mapping = aes(x = displ, y = hwy))

### Solutions

1. You should see a blank plot. This is what ggplot does, it simply creates a “canvas” to which you can add “layers”.
2. How many rows are in mpg? How many columns?

str(mpg)

## tibble [234 × 11] (S3: tbl\_df/tbl/data.frame)  
## $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...  
## $ model : chr [1:234] "a4" "a4" "a4" "a4" ...  
## $ displ : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...  
## $ year : int [1:234] 1999 1999 2008 2008 1999 1999 2008 1999 1999 2008 ...  
## $ cyl : int [1:234] 4 4 4 4 6 6 6 4 4 4 ...  
## $ trans : chr [1:234] "auto(l5)" "manual(m5)" "manual(m6)" "auto(av)" ...  
## $ drv : chr [1:234] "f" "f" "f" "f" ...  
## $ cty : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...  
## $ hwy : int [1:234] 29 29 31 30 26 26 27 26 25 28 ...  
## $ fl : chr [1:234] "p" "p" "p" "p" ...  
## $ class : chr [1:234] "compact" "compact" "compact" "compact" ...

There are 234 observations of 11 variables. thus, there are 11 columns and 234 rows. we can confirm this by looking at the dimension of the underlying data matrix:

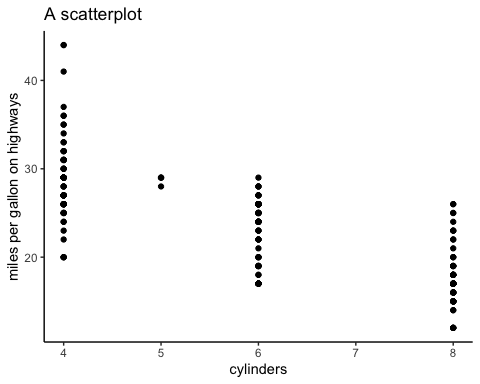
dim(mpg)

## [1] 234 11

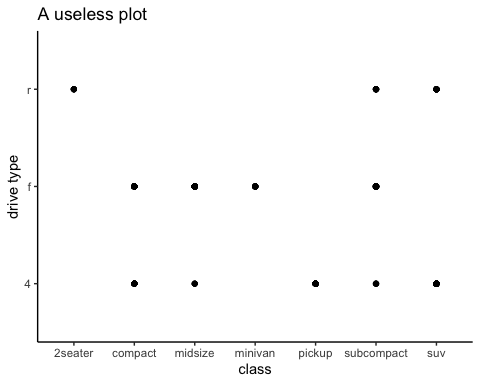
?mpg

*drv* is a categorical variable indicating the drive type: f = front-wheel drive, r = rear wheel drive, 4 = 4wd

ggplot(data = mpg) +   
 geom\_point(mapping = aes(y = hwy,x = cyl)) +   
 theme\_classic() +   
 labs(title = "A scatterplot", y = "miles per gallon on highways",x="cylinders")

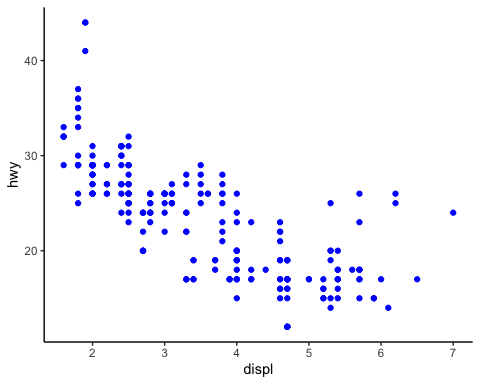


ggplot(data = mpg) +   
 geom\_point(mapping = aes(x = class,y = drv)) +   
 theme\_classic() +   
 labs(title = "A useless plot", x = "class",y="drive type")

 both *drv* and *class* are categorical variable and it does not make sense to visualize them this way. What would be a better way to show these data?

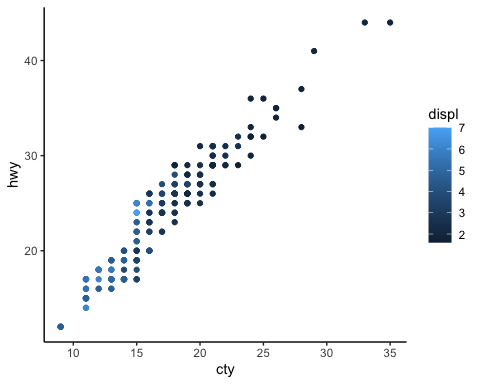
1. Color is used within the aesthetics function, where we specify which variables should be used. “blue” is however not a variable in our data frame. Thus it is ignored. The following code creates the correct plot:

ggplot(data = mpg) +   
 geom\_point(mapping = aes(x = displ, y = hwy), color = "blue")



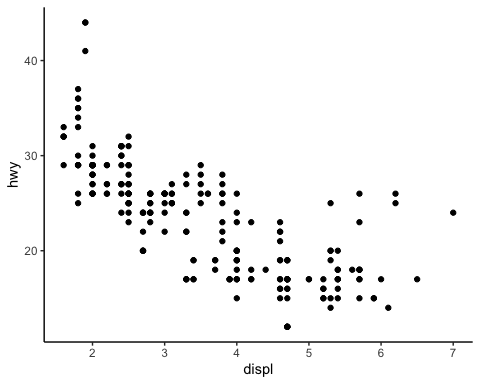
A continous variable will lead ot a continous color gradient rather than a discrete set of colors.

ggplot(data = mpg) +   
 geom\_point(mapping = aes(x = cty, y = hwy, color = displ))



1. The “+” symbol always has to be in the top row:

ggplot(data = mpg) +   
 geom\_point(mapping = aes(x = displ, y = hwy))



## Week 3 (10.03.2021)

#### Exercise 7

Load the data set “chap04e1HumanGeneLengths.csv” (the lengh of genes in the human genome, found on ilias) and answer the following questions and provide the R code you used for that. Put all of your analysis in a single R script.

1. How many individuals are in the sample (i.e., what is the sample size, n)?
2. What is the sum of all of the observations?
3. What is the mean of this sample?
4. What is the sum of the squares of the measurements?
5. What is the variance of this sample?
6. What is the standard deviation of this sample?
7. What is the coefficient of variation for this sample?
8. Display the data using different plotting techniques. Which one illustrates the data best?

#### Solution

Load the data

genes <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter04/chap04e1HumanGeneLengths.csv"))

1. How many individuals are in the sample (i.e., what is the sample size, n)?

n = dim(genes)[1]  
print(n)

## [1] 20290

1. What is the sum of all of the observations?

sum.gene.lengths = sum(genes$geneLength)

1. What is the mean of this sample?

mean.gene.length1 = sum.gene.lengths/n  
mean.gene.length2 = mean(genes$geneLength)  
  
print(mean.gene.length1)

## [1] 2622.027

print(mean.gene.length2)

## [1] 2622.027

1. What is the sum of the squares of the measurements?

square.sum.gene.lengths = sum(genes$geneLength^2)  
print(square.sum.gene.lengths)

## [1] 223678143206

1. What is the variance of this sample?

# by hand:  
variance.gene.length = sum((genes$geneLength - mean.gene.length1)^2)/(n-1)  
  
print(variance.gene.length)

## [1] 4149235

# for comparison:  
var(genes$geneLength)

## [1] 4149235

1. What is the standard deviation of this sample?

sd(genes$geneLength)

## [1] 2036.967

# or   
  
sd.gene.length = sqrt(variance.gene.length)

1. What is the coefficient of variation for this sample?

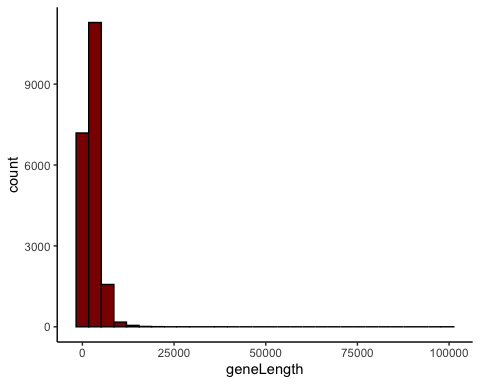
cv.gene.length = sd.gene.length/mean.gene.length1  
print(cv.gene.length)

## [1] 0.7768672

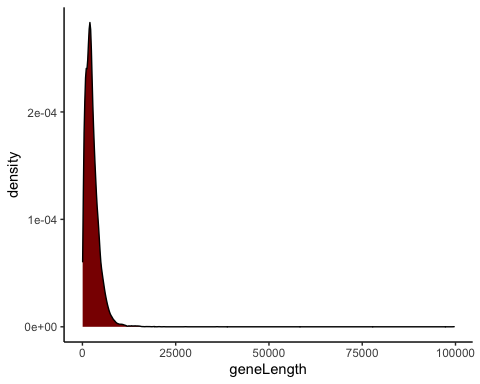
1. Display the data using different plotting techniques. Which one illustrates the data best?

# Which one do you like best?  
  
ggplot(data = genes) +   
 geom\_histogram(mapping = aes(x = geneLength),color="black",fill="darkred") +   
 theme\_classic()

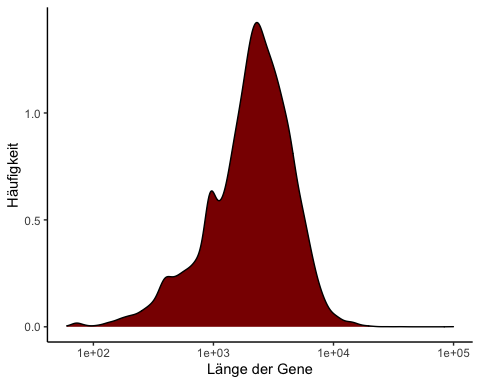
## `stat\_bin()` using `bins = 30`. Pick better value with `binwidth`.



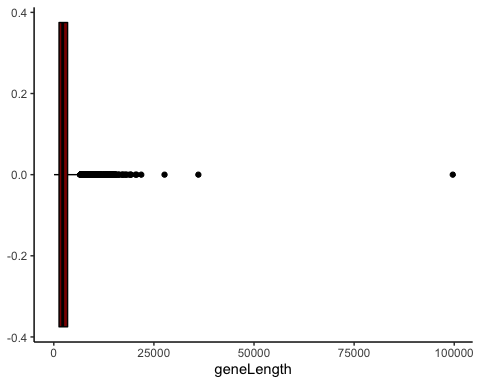
ggplot(data = genes) +   
 geom\_density(mapping = aes(x = geneLength),color="black",fill="darkred") +   
 theme\_classic()



ggplot(data = genes) +   
 geom\_density(mapping = aes(x = geneLength),color="black",fill="darkred") +   
 xlab("Länge der Gene") +  
 ylab("Häufigkeit") +  
 scale\_x\_log10() +  
 theme\_classic()



ggplot(data = genes) +   
 geom\_boxplot(mapping = aes(y = geneLength),color="black",fill="darkred") +   
 theme\_classic() +   
 coord\_flip()



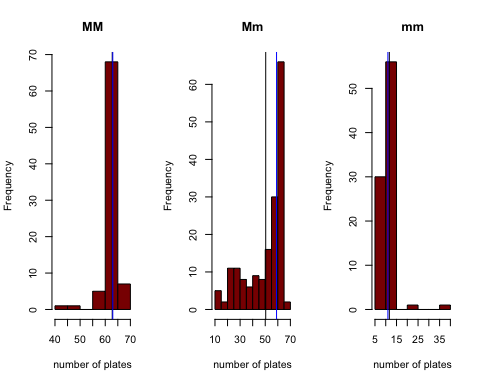
## Week 4 (17.03.2021)

#### Exercise 8

Use the stickleback dataset and calcualte the median and mean for each genotpye. Make a histogramm and add vertical lines to the histogramm indicating the mean and median for each genotype. What do you see? How does the shape of the distribution affect differences between mean and median?

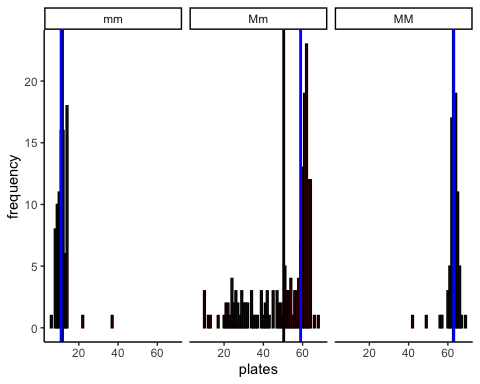
##### Solution

# load the data file:  
stickleback <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter03/chap03e3SticklebackPlates.csv"))  
  
# one possible solution:  
  
mean.mm = mean(stickleback$plates[stickleback$genotype=="mm"])  
mean.mM = mean(stickleback$plates[stickleback$genotype=="Mm"])  
mean.MM = mean(stickleback$plates[stickleback$genotype=="MM"])  
  
median.mm = median(stickleback$plates[stickleback$genotype=="mm"])  
median.mM = median(stickleback$plates[stickleback$genotype=="Mm"])  
median.MM = median(stickleback$plates[stickleback$genotype=="MM"])  
  
par(mfrow = c(1,3))  
  
hist(stickleback$plates[stickleback$genotype=="MM"],  
 main = "MM",xlab = "number of plates",  
 col="darkred")  
  
abline(v = mean.MM,col="black")  
abline(v = median.MM,col="blue")  
  
hist(stickleback$plates[stickleback$genotype=="Mm"],  
 main = "Mm",xlab = "number of plates",  
 col="darkred")  
abline(v = mean.mM,col="black")  
abline(v = median.mM,col="blue")  
  
hist(stickleback$plates[stickleback$genotype=="mm"],  
 main = "mm",xlab = "number of plates",  
 col="darkred")  
abline(v = mean.mm,col="black")  
abline(v = median.mm,col="blue")



# alternatively, we can first create a data frame   
# with the means per genotype  
# check out the help function of ddply  
# (this is only one way to do this, you can also do it by   
# "hand" or using other functions such as tapply) :  
  
library(dplyr)  
df.mean = ddply(stickleback, "genotype", summarize, m.number = mean(plates))  
df.median = ddply(stickleback, "genotype", summarize, m.number = median(plates))  
  
# then we plot it with a single ggplot  
ggplot() +  
 geom\_bar(data = stickleback, mapping = aes(x=plates),binwidth=5,color="black",fill="darkred") +  
 geom\_vline(data = df.mean, aes(xintercept=m.number),col="black",size=1) +  
 geom\_vline(data = df.median, aes(xintercept=m.number),col="blue",size = 1) +  
 facet\_wrap(~ genotype) +  
 ylab("frequency") +  
 theme\_classic()

## Warning: Ignoring unknown parameters: binwidth



#### Exercise 9 (standard)

Copy and paste this R code into a script. Before you run it, try to figure out what it does. Modify the output so that it looks nicer, i.e., add labels, change theme, change colors, …

Tip 1: Use the help function or the internet to figure out what the different commands and functions are doing.

Tip 2: This might also be helpful: <https://s-peischl.github.io/StatisticsForBiology/#Working_with_Distributions_and_random_variables>

Tip 3: Look at the next exercise, it may give you an idea what is happening in this R code …

reps = 1000  
sample\_size = 10  
mu = 10  
sigma = 5  
means = vector("numeric",reps)  
  
for (i in 1:reps)  
{  
 sample = rnorm(sample\_size,mu,sigma)  
 means[i] = mean(sample)  
}  
  
  
  
x = seq(mu-2\*sigma,mu+2\*sigma,by=0.01)  
  
SE = sigma/sqrt(sample\_size)  
  
y = dnorm(x,mu,SE)  
  
df2 = data.frame(x = x,y = y)  
  
df = data.frame(means = means)  
ggplot(data = df) +   
 geom\_histogram(aes(x = means),binwidth=1) +  
 geom\_line(data = df2,aes(x = x,y = y\*reps))

##### Solution

This script simulates the sampling distribution of the mean.

#### Exercise 9 (challenging)

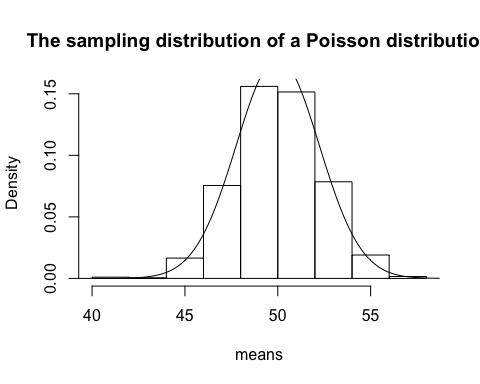
Write a short R script that simulates the sampling distribution of the mean. Plot the sampling distribution and add the theoretical expectation to the plot. Follow the following steps:

1. Choose a distribution that represents your population (e.g., Normal or Poisson).
2. Draw a sample of n individuals from that population (hint: use rnorm, rpoiss, etc.. to draw a random sample), and store it in a vector.
3. Calculate the mean of the sample. Store the mean of this sample in a vector.
4. Do this 1000 times, so that you have calculated a 1000 means from a 1000 samples. Hint: use a for-loop to automate this.
5. Plot a histogram of the sample means.
6. Add a line for the theoretical expectation of the sampling distribution. Hint: use the function dnorm to get the density of the normal distribution.

Tip if you get stuck: Have a look at the previous exercise for inspiration …

##### Solution

# We choose the Poisson distribution with mean lambda = 50  
lambda = 50  
  
# We do 1000 replicates  
reps = 1000  
  
# We choose the sample size n to be 10  
sample\_size = 10  
  
# the means of each sample will be stroed in this vector  
means = vector("numeric",reps)  
  
# in this loop, we will calcualte the means of the sample  
for (i in 1:reps)  
{  
 # take a sample of the poisson distribution  
 sample = rpois(sample\_size,lambda)  
   
 # store the mean in our vector  
 means[i] = mean(sample)  
}  
  
# a histogram - note that we set freq = FALSE to   
# show realtive frequencies   
hist(means,freq=F,main="The sampling distribution of a Poisson distribution")  
  
# we specify the x values for the theoretical poisson  
x = seq(0,3\*lambda,by=0.1)  
  
#we calcualte the standard error:  
SE = sqrt(lambda)/sqrt(sample\_size)  
  
# caclualte the corresponding PDF of the poisson  
y = dnorm(x,lambda,SE)  
  
lines(x,y)



## Week 5 (24.03.2021)

### Exercise 10

The pizza below, ordered from the Venn Pizzeria on Bayes Street, is divided into eight slices.

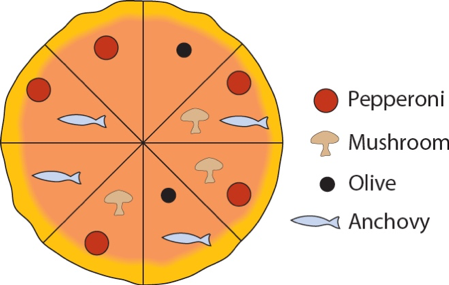


Figure: Venn Pizza

Answer the following questions based on the drawing of the pizza:

1. What is the probability that a randomly drawn slice has pepperoni on it?
2. What is the probability that a randomly drawn slice has both pepperoni and anchovies on it?
3. What is the probability that a randomly drawn slice has either pepperoni or anchovies on it?
4. Are pepperoni and anchovies mutually exclusive on this pizza?
5. Are olives and mushrooms mutually exclusive on this pizza?
6. Are getting mushrooms and getting anchovies independent when randomly picking a slice of pizza?
7. If I pick a slice from this pizza and tell you that it has olives on it, what is the chance that it also has anchovies?
8. If I pick a slice from this pizza and tell you that it has anchovies on it, what is the chance that it also has olives?
9. Seven of your friends each choose a slice at random and eat them without telling you what toppings they had. What is the chance that the last slice has olives on it?
10. You choose two slices at random. What is the chance that they have both olives on them? (Be careful: after removing the first slice, the probability of choosing one of the remaining slices changes.)
11. What is the probability that a randomly chosen slice does not have pepperoni on it?
12. Draw a pizza for which mushrooms, olives, anchovies and pepperoni are all mutually exclusive

#### Solution

The pizza below, ordered from the Venn Pizzeria on Bayes Street, is divided into eight slices.

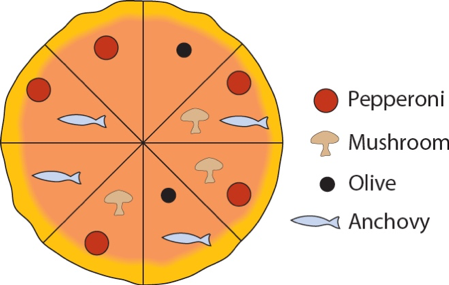


Figure: Venn Pizza

Answer the following questions based on the drawing of the pizza:

1. What is the probability that a randomly drawn slice has pepperoni on it?
2. What is the probability that a randomly drawn slice has both pepperoni and anchovies on it?
3. What is the probability that a randomly drawn slice has either pepperoni or anchovies on it?
4. Are pepperoni and anchovies mutually exclusive on this pizza?

No. There are two slices with both pepperoni and anchovies.

1. Are olives and mushrooms mutually exclusive on this pizza?

Yes. There is no slice that has both olives and mushrooms on it.

1. Are getting mushrooms and getting anchovies independent when randomly picking a slice of pizza?

No, because

and

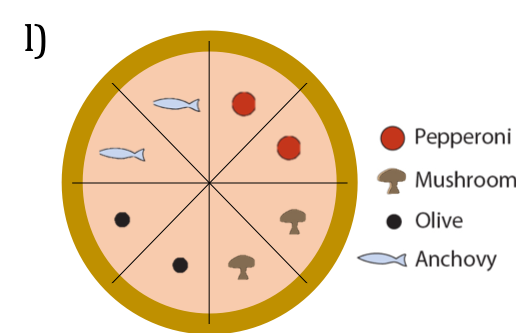
but

1. If I pick a slice from this pizza and tell you that it has olives on it, what is the chance that it also has anchovies?
2. If I pick a slice from this pizza and tell you that it has anchovies on it, what is the chance that it also has olives?
3. Seven of your friends each choose a slice at random and eat them without telling you what toppings they had. What is the chance that the last slice has olives on it?

.

This can be seen directly because each slice has the same probability to be the last slice.

1. You choose two slices at random. What is the chance that they have both olives on them? (Be careful: after removing the first slice, the probability of choosing one of the remaining slices changes.)
2. What is the probability that a randomly chosen slice does not have pepperoni on it?
3. Draw a pizza for which mushrooms, olives, anchovies and pepperoni are all mutually exclusive



A pizza for which mushrooms, olives, anchovies and pepperoni are all mutually exclusive

### Exercise 11

After graduating from your university with a biology degree, you are interviewed for a lucrative job as a snake handler in a circus sideshow. As part of your audition, you must pick up two rattlesnakes from a pit. The pit contains eight snakes, three of which have been defanged and are assumed to be harmless, but the other five are definitely still dangerous. Unfortunately budget cuts have eliminated the herpetology course from the curriculum and so you have no way of telling in advance which snakes are dangerous and which are not. You pick one snake with your left hand and one with your right.

1. What is the probability that you picked up no dangerous snakes?
2. Assume that any dangerous snake that you pick up has a probability of 0.8 of biting you. The defanged snakes do not bite. What is the chance that, in picking up your two snakes, you are bitten at least once?
3. If you picked up one snake and it didn???t bite you, what is the probability that it is defanged.

#### Solution

1. What is the probability that you picked up no dangerous snakes?
2. Assume that any dangerous snake that you pick up has a probability of 0.8 of biting you. The defanged snakes do not bite. What is the chance that, in picking up your two snakes, you are bitten at least once?

(from (a))

Putting al this together:

1. If you picked up one snake and it didn’t bite you, what is the probability that it is defanged?

$$P(no \, bite) = P(defanged)P(no bite \mid defanged) $$  
 $$+ P(dangerous)P(no \, bite \mid dangerous) = (3/8 \*1) + (5/8 (1-0.8)) = 0.5$$  
   
 Therefore   
 $$P(defanged \mid no \, bite) = (1\*3/8 )/ (0.5) = 0.75 = 75\%.$$

### Exercise 12

Five different researchers independently take a random sample from the same population and calculate a 95% confidence interval for the same parameter.

1. What is the probability that all five researchers have calculated an interval that includes the true value of the parameter?
2. What is the probability that at least one does not include the true parameter value.

#### Solution

1. What is the probability that all five researchers have calculated an interval that includes the true value of the parameter?
2. What is the probability that at least one does not include the true parameter value.

## Week 6 (31.03.2021)

### Exercise 13

1. Make a plot of the probability density funciton of a Normal distribution. Choose values for the mean and variance. Next make a plot of the probability density function of the standard Normal distribution. What do you see? How are these plots different / similar?
2. Draw a random sample (sample size ) from the Normal distribution that you chose in (a). Plot a histogram of that sample.
3. Combine the two plots in a single plot, i.e., superimpose the probaility density function on the histogram.
4. Calculate the 10 percent quantile of the distribution. Store the value in a variable . Then calcualte the proportion of numbers in your random sample from (b) that are smaller or equal to . What do you find?

### Exercise 14

1. Let be a random variable that follows the standrad normal distribution. Calculate the percentiles of the standard Normal distribution. Store the values in a vector . Next calculate the cummulative distribution function for the values in . Store this in a variable . Finally, plot against . What do you see? Can you explain?
2. [advanced] Try to describe the finding from (a) using mathematical reasoning.

### Exercise 15

Plot the probability density function of a Normal distribution.

1. Indicate the smallest 5 percent of the distribution. In other words: Let be a normally distributed random variable. For which range to we get ?
2. Indicate the most extreme 5 percent of the distribution. In other words: For which range to we get ? [Hint: you can use the quantile function of and the symmetry of the Normal distribution to simplify things]

## Week 7 (07.04.2021)

#### Exercise 16

1. Use the stickleback data set. Calculate a confidence interval for the mean plate numbers for each genotype. Make a plot that shows the mean and the confidence interval of each genotype.
2. Now perform a t.test for each comparison of plate numbers between genotypes. Compare the outcome of the t.tests with the results from the confidence intervals. What do you find?

#### Exercise 17

Use the data set “chap11e2Stalkies.csv” of using eye span measurements from a sample of stalk-eyed flies. Perform a one sample t-test by “hand”, that is, clacualte the t statistic and the p-value in R, without using the function t.test (or a similar function). Test the null hypothesis that the mean eye span is equal to . Use a significance threshold of 0.05 and a two-sided test. Then check your results with the function t.test. Make a plot to illustrate your analysis.

## Week 8 (21.04.2021)

Q & A and repetitorium

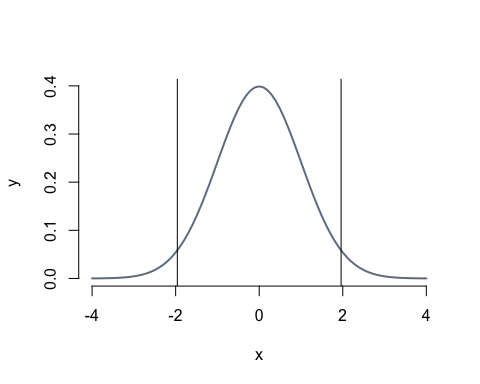
### Student Question 1

What are the in the formula for confidence intervalls ? What is the onfidence level? Why to you use and not ?

### Answer

The formula can be found on page 159 of the lecture slides. In a nutshell, we use the width of the sampling distribution to tell us how wide our confidence intervall should be. For this we need to calcluate the quantiles of the sampling distribution. For large samples, this can be approximated by a normal distribution. This is where value comes into play: it is the percentile of the standard normal distribution. This means that it is the value that delimits the smallest of the distribution from the largest . We call $ 1- $ the confidence level. Note that , thus we calcualte the range in which the most common observation lie. The quantiles cannot be calcualted straightforwardly by hand, but R can solve this for us. The following plot illustrates the quantiles.

mu = 0  
sigma = 1   
x = seq(mu-4\*sigma,mu+4\*sigma,by=0.1)  
y = dnorm(x, mean = mu, sd = sigma)  
  
plot(x,y,type="l",col="slategray",lwd=2,axes=F)  
axis(1)  
axis(2)  
  
alpha = 0.05  
  
  
z1 = qnorm(alpha/2,mu,sigma)  
z2 = qnorm(1-alpha/2,mu,sigma)  
  
abline(v=c(z1,z2))



### Student Question 2

Why do we always set y as the dnorm() of our x values when we plot a normal distribution?

### Answer

Because the function dnorm gives us the density function of a noraml distribution. You can try using the hepl function to see what a function does:

?dnorm

### Student Question 3

I want to define e.g. “N” as the number of numeric values that are in column 3 of my dataframe. So if my dataframe has 9 numeric values in column 3, how can I set N = 9 without typing N = 9, but rather N = “number of numeric values in column 3 of my data frame”? This might be useful if a column has 345 values and I don’t want to count them or without using the str() function to actually show me the number. I could only find one solution and that is using the function “length()”…but is there another way?

### Answer

The function *length()* is a good option. For dataframes, you can alos use *dim()* to get the number of rows and columns of the dataframe.

### Student Question 4

Exercise 14: Could you explain again what you did in Exercise 14. Why did you need to put a vector of probabilities into the qnorm function and what does this qnorm function then do with this vector of probabilities in detail? And maybe also explain again why pnorm is the inverse of qnorm; also how this is shown with mathematical reasoning? This was not clear in the exercise lecture/podcast!

### Answer

If I pass a vector of probailites to the function qnorm, eg, qnorm(c(0.1,0.2,0.3)), it will return a vector with the 0.1, 0.2 and 0.3 percentile.

qnorm(c(0.1,0.2,0.3))

## [1] -1.2815516 -0.8416212 -0.5244005

The quantile function and the cumulative distirbution function are by defnition inverse functions. is the p-percentile. Then gives you the value such that . Conversly, the cumulative distirbution function is given by . Alternatively, one can see that the percentile satisifies .

### Student Question 5

Exercise 17: Why did you calculate the p-value wit this “-abs”-part? And is it also fine to just type "2\*pt(test.statistic, df = 8)"?

### Student Question 6

Exercise 17: Why did you add the “-9” in the command of the t-test (t.test(stalkie$eyespan-9))? I don’t get this “-9”-part when looking at the help page on R.

### Answer

By default the one sample t.test() function test the null hypotehsis . If i substract 9 from each data point i test the nul hypotehsis . A better way would be to specify the null hypothesis in the t.test function via the parameter mu:

t.test(x, mu = 9)

### Student Question 7

Exercise 17: In the description of the exercise you write that we should use a threshold value alpha of 0.05. How do we include this in our calculations? Or if it is already a standard setting in one of our functions we used, how would you proceed if we had to choose a threshold value alpha of 0.1?

### Answer

The threshold value is only used for the interpreation part here. If the P-value is smaller than the threshold, you reject the null hypothesis. Otherwise you don’t.

### Student Question 8

And last but not least a more general question: How do you proceed when you have an exercise like e.g. exercise 16a)? I mean I did not get that I would have to make a dataframe of the means, lower and upper confidence intervalls and genotypes at first…Do you e.g. first imagine yourself how the plot should look like, or how do you know what to do?

### Answer

There are many ways to do such plots and I usually proceed like this:

* make a sketch of your plot (on paper or in your mind)
* think what information you need for such a plot
* find a function that you want to use and look up how it works
* organize your data so that you can “feed” it to the functions you want to use

For ggplot functions, this means organizing yur data in a data frame. In base R this is often not necessary.

## Week 9 (28.04.2021)

### Exercise 18

Perform a goodness-of-fit test for a data set with two categories. The data file is “chap08e4XGeneContent.csv” and can be found on ilias. Start by reading and inspecting the data. Each row is a different gene, with its occurrence on the X chromosome indicated. Then test if the proportion of genes on the X chromsome (among all genes in the genome) is the same as the proportional length of the X chromosome measured in base pairs (among the whole genome). The relative size of the X chromosome is given by and that of the other chromomoses is .

1. Plot a frequency table showing the number of genes on the X and on other chromosomes.
2. Perform a goodness-of-fit test to the proportional model.
3. Draw a conclusion from your test.

### Exercise 19

Load the dataset chap08e1DayOfBirth.csv. Each row represents a single birth, and shows the day of the week of birth. Test the null hypothesis that birthdays are randomly distributed in the sense that each day occurs with the same probability. Visualize your results and give an interpretation of the test results. Can you come a up with a potential explanation?

## Week 10 (05.05.2021)

### Exercise 20

Assume that Z is a number randomly chosen from a standard normal distribution. Calculate each of the following probabilities : a) P(Z > 1.34) b) P(Z < 1.34) c) P(Z < 2.15) d) P(Z < 1.2) e) P(0.52 < Z < 2.34) f) P(-2.34 < Z < -0.52) g) P(Z < -0.93) h) P(-1.57 < Z < - 0.32)

Make a sketch for each case and calculate the values using R functions.

### Exercise 21

As the world warms, the geographic ranges of species might shift toward cooler areas. Chen et al. (2011) studied recent changes in the highest elevation of 31 taxa, in meters, over the late 1900s and early 2000s. Positive numbers indicate upward shifts in elevation, and negative numbers indicate shifts to lower elevations. The values are given by:

data = c(58.9, 7.8, 108.6, 44.8, 11.1, 19.2, 61.9, 30.5, 12.7, 35.8, 7.4, 39.3, 24.0, 62.1, 24.3, 55.3, 32.7, 65.3, -19.3, 7.6, -5.2, -2.1, 31.0, 69.0, 88.6, 39.5, 20.7, 89.0, 69.0, 64.9, 64.8)

1. What is the sample size n?
2. What is the mean of these data points? Remember to give the units.
3. What is the standard deviation of elevational range shift? (Give the units as well.)
4. What is the standard error of the mean for these data?
5. How many degrees of freedom will be associated with a confidence interval and a one-sample t-test for the mean elevation shift?
6. Calculate the 95 % confidence interval for the mean using these data.
7. For the one-sample t-test, write down the appropriate null and alternative hypotheses.
8. Calculate the test statistic t for this test.
9. What assumptions are necessary to do a one-sample t-test?
10. Calculate the P-value for this test as accurately as you can.
11. Did species change their highest elevation on average?

### Exercise 22

A four-year review at the provincial Hospital in Alotau, Papua New Guinea (Barss 1984), found that 1/40 of their hospital admissons were injuries due to falling coconuts. If coconuts weigh on average 3.2 kg, and the upper bound of the 95% CI is 3.5 kg, what is the lower bound of this CI? Assume a normal distribution of coconut weights.

## Week 11 (12.05.2021)

### Exercise 23

A forensic pathologist wants to know whether there is a difference between the rate of cooling of freshly killed bodies and those which were reheated, to determine whether you can detect an attempt to mislead a coroner about time of death. He tested several mice for their “cooling constant” both when the mouse was originally killed and then after the mouse was re-heated. The data can be found in the file “mouse\_cooling.csv” on ilias. Test whether there is any difference in the cooling constants between freshly killed and reheated corpses. Check whether you can use a t-test and justify your decision to use it. If a t-test is notjsutified, try to use an alternative strategy (e.g., non-parametric test).

### Exercise 24

For each of the following scenarios, identify the best statistical test to use and state the null hypothesis. (Please note, do not give the answer to the specific question, but simply state the best test to use and the null hypothesis for the scenario.)

1. Asking whether stickleback fish occur with equal probability through all areas of a pond. We have raken random samples at various locations of the pond (all the same size) and have recorded the number of fish in each sampled location.
2. We want to know which of two tumor treatments showed better results. We recorded the change in tumor size during the treatment in two groups of patients, group 1 was treated with drug A and group 2 with drug B.
3. We want to know whether a new tumor treatment showed good results. We recorded the change in tumor size before and after the treatment in each patient.
4. Asking whether the number of smokers in a list of cities is proportional to the population size of those cities.
5. Testing whether patients change in body mass during a hospital stay.

### Exercise 25

Health spending per person from a random sample of 20 countries is given in the file “healt\_expenditure.csv” on ilias. We will use this sample to estimate the mean of log health expenditure, including a confidence interval.

1. Visualize the frequency distribtuion using a histogram. Does the data deviate from a normal distribution? If yes, how?
2. Why is a log transformation a good choice for these data?
3. What is the sample size?
4. Calcualte the natural logarithm of each data point.
5. What is the mean log health expenditure?
6. What is the standard deviation of the log health expenditure?
7. Calculate the standard error of the mean log health expenditure?
8. Calculate the 95% confidence interval of the mean log health expenditure.
9. What are the limits of this confidence interval expressed on original (non-log) scale?

## Week 12 (19.05.2021)

### Exercise 26

Recycling paper has some obvious benefits, but it may have unintended consequences. For example, perhaps people are less careful about how much paper they use if they know their waste will be recycled. Catlin and Wang (2013) tested this idea by measuring paper use in two groups of experimental participants. Each person was placed in a room alone with scissors, paper and a trash can, and was told that he or she was testing the scissors. In the “recycling” group only, there was also a recycling bin in the room. The amount of paper used by each participants was measured in grams.

No recycling bin: 4,4,4,4,4,4,4,5,8,9,9,9,9,12,12,13,14,14,14,14,15,23 With recycling bin: 4,5,8,8,8,9,9,9,12,14,14,15,16,19,23,28,40,43,129,130

1. Make and examine a histogram of these data. Are the frequency distributions of the two groups similar in shape and spread?
2. Based on (a), discuss your options for testing a difference between the groups.
3. Apply a Mann-Whitney U-test (R function *wilcox.test*). State the null and alternative hypotheses.
4. Calculate the P-value as accurately as you can and interpret the result.

### Exercise 27

Use the data from exercise 26 and perform permutation test on the data. Visulaize the empirical sampling distribution and calcualte a P-value. Compare it to the results fromthe exercise 26. Hint: Use a for loop for the permutations. Alternatively, take a look at the package: *perm* and the function *permTS*.

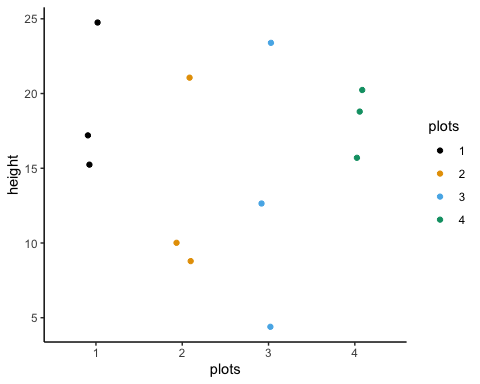
## Week 13 (26.05.2021)

### Exercise 28

A plant ecologist wishes to test the hypothesis that the height of species X depends on the type of soil it grows in. She measures the height of 3 plants in each of 4 plots representing 4 different soil types. The results are tabulated below. (Height in centimeters.) Do the results support her hypothesis, assuming normality and equality of variances?

The data is

height = c(15,25,17,10,9,21,23,13,4,19,20,16)  
plots = rep(c(1,2,3,4),each=3)  
  
data = data.frame(height = height,plots = as.factor(plots))  
  
ggplot(data,aes(x=plots, y=height,col=plots)) +   
 geom\_jitter(position=position\_jitter(0.1))



#### Solution

model = lm(height ~ plots, data = data)  
anova(model)

## Analysis of Variance Table  
##   
## Response: height  
## Df Sum Sq Mean Sq F value Pr(>F)  
## plots 3 86 28.667 0.6866 0.585  
## Residuals 8 334 41.750

We cannot conclude that the different soild types affect the height.

### Exercise 29

Test a linear correlation between inbreeding coefficients of litters of mated wolf pairs and the number of pups surviving their first winter.

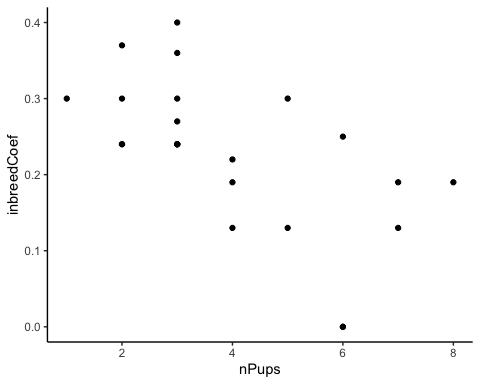
wolf <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter16/chap16e2InbreedingWolves.csv"))  
head(wolf)

## inbreedCoef nPups  
## 1 0.00 6  
## 2 0.00 6  
## 3 0.13 7  
## 4 0.13 5  
## 5 0.13 4  
## 6 0.19 8

1. Calculate the correlation coefficient.
2. Perform a test of the null hyptohesis that . Do all calcualtions “by hand”.
3. Perform the test with the function *cor.test*.

#### Solution

ggplot(data= wolf, aes(x = nPups,y=inbreedCoef))+  
 geom\_point()



cor(wolf$inbreedCoef,wolf$nPups)

## [1] -0.6077184

1. and c)

r = cor(wolf$inbreedCoef,wolf$nPups)  
n = length(wolf$inbreedCoef)  
SE = sqrt((1-r^2)/(n-2))  
t = r/SE  
t

## [1] -3.5893

p.value = 2\*pt(t,df=n-2)  
  
cor.test(wolf$inbreedCoef,wolf$nPups)

##   
## Pearson's product-moment correlation  
##   
## data: wolf$inbreedCoef and wolf$nPups  
## t = -3.5893, df = 22, p-value = 0.001633  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.8120418 -0.2706791  
## sample estimates:  
## cor   
## -0.6077184

### Exercise 30

Test the null hypothesis of zero regression slope. The data are from an experiment investigating the effect of plant species diversity on the stability of plant biomass production.

Data:

plant <- read.csv(url("http://www.zoology.ubc.ca/~schluter/WhitlockSchluter/wp-content/data/chapter17/chap17e3PlantDiversityAndStability.csv"))  
head(plant)

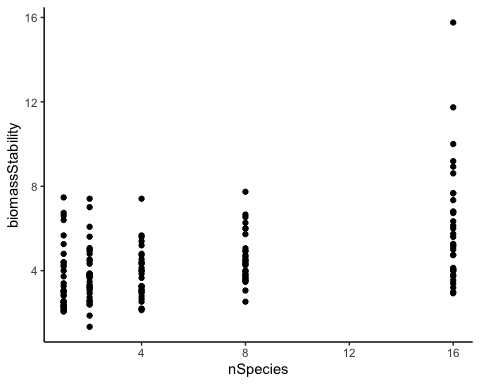
## nSpecies biomassStability  
## 1 1 7.47  
## 2 1 6.74  
## 3 1 6.61  
## 4 1 6.40  
## 5 1 5.67  
## 6 1 5.26

1. Inspect the data. Would a log trandform make sense?
2. Make a scatter plot of the data. Add a regression line.
3. Test if the null hypothesis of a slop of zero can be rejected.
4. Write down the model with the estimated parameters.

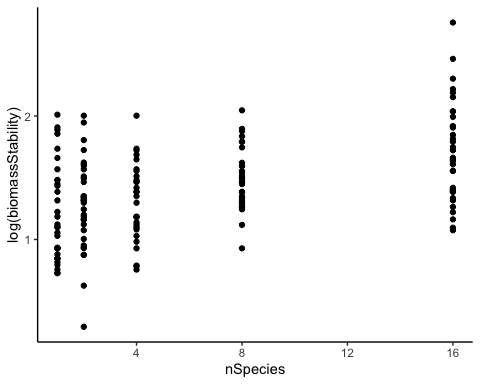
#### Solution

1. Inspect the data. Would a log trandform make sense?

ggplot(data= plant,aes(x = nSpecies,y= biomassStability))+  
 geom\_point()

 We see that the variance is increasing with increasing number of species. A log transform seems appropriate.

ggplot(data= plant,aes(x = nSpecies,y= log(biomassStability)))+  
 geom\_point()

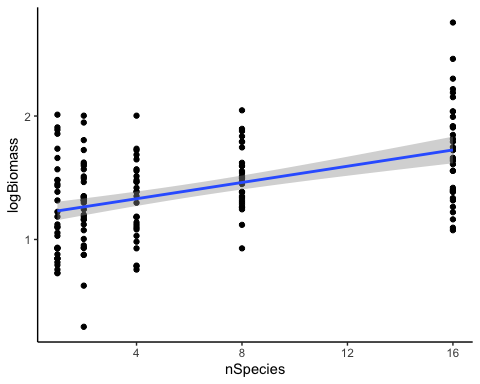
 Note the change in the y axis scale. The varaince is now much more similar across values for nSpecies.

plant$logBiomass = log(plant$biomassStability)

1. Make a scatter plot of the data. Add a regression line.

ggplot(data= plant,aes(x = nSpecies,y= logBiomass))+  
 geom\_point() +   
 geom\_smooth(method = "lm")

## `geom\_smooth()` using formula 'y ~ x'



1. Test if the null hypothesis of a slop of zero can be rejected.

model =lm(logBiomass~nSpecies,data=plant)  
summary(model)

##   
## Call:  
## lm(formula = logBiomass ~ nSpecies, data = plant)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.97148 -0.25984 -0.00234 0.23100 1.03237   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.198294 0.041298 29.016 < 2e-16 \*\*\*  
## nSpecies 0.032926 0.004884 6.742 2.73e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3484 on 159 degrees of freedom  
## Multiple R-squared: 0.2223, Adjusted R-squared: 0.2174   
## F-statistic: 45.45 on 1 and 159 DF, p-value: 2.733e-10

We find that the p value is extremely small, we therefore can reject the null hypothesis of .

1. Write down the model with the estimated parameters.

beta0 = model$coefficients[1]  
beta1 = model$coefficients[2]  
sigma = sd(model$residuals)  
sigma

## [1] 0.3472932

our model is thus

with

beta0

## (Intercept)   
## 1.198294

beta1

## nSpecies   
## 0.0329259

and

is drawn from a normal distribution with mean 0 and standard deviation

sigma

## [1] 0.3472932