BIO4158 Applied biostats with R

Laboratory manual

Julien Martin

16-11-2021

Table des Matières

Note

Development version. Lab material will appear slowly during the Fall 2021 term.

Preface

The laboratory exercises outlined in the following pages are designed to allow you to develop some expertise in using statistical software (R) to analyze data. R is powerful statistical software but, like all software, it has its limitations. In particular, it is dumb: it cannot think for you, it cannot tell you whether the analysis you are attempting to do is appropriate or even makes any sense, and it cannot interpret your results.

General points to keep in mind

- Before attempting any statistical procedure, you must familiarize yourself with what the procedure is actually doing. This does not mean you actually have to know the underlying mathematics (although this certainly helps!), but you should at least understand the principles involved in the analysis. Therefore, before doing a laboratory exercise, read the appropriate section(s) in the lecture notes. Otherwise, the output from your analyses - even if done correctly - will seem like drivel.
- The laboratories are designed to complement the lectures, and vice versa. Owing to scheduling constraints, it may not be possible to synchronize the two perfectly. But feel free to bring questions about the laboratories to class, or questions about the lectures to the labs.
- Work on the laboratories at your own speed: some can be donemuch more quickly than others, and one laboratory need not correspond to one laboratory session. In fact, for some laboratorieswe have allotted two laboratory sessions. Although you will notbe "graded" on the laboratories per se, be aware that completingthe labs is essential. If you do not complete the labs, it is veryunlikely that you will be able to complete the assignments and thefinal exam/term paper. So take these laboratories seriously!
- The objective of the first lab is to allow you to acquire or reviewthe minimum knowledge required to complete the following laboratory exercises with R. There are always several methods toaccomplish something in R, but you will only find simple ways inthis manual. Those amongst you that want to go further will easilyfind many examples of more detailed and sophisticated methods.In particular, I point you to the following resources:
  - R for beginners http://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf
  - An introduction to R http://cran.r-project.org/doc/manuals/R-intro.html
  - If you prefer paper books, the CRAN web site has a commented list at : http://www.r-project.org/doc/bib/R-books.html
  - Excellent list of R books https://www.bigbookofr.com/

- R reference card by Tom Short http://cran.r-project.org/doc/contrib/Short-refcard.pdf

What is R and why use it in this course?

R is multiplatform free software forming a system for statistical computation and graphics. R is also a programming language specially designed for statistical data analysis. It is a dialect of the S language. S- Plus is another dialect of the S language, very similar to R, incorporated into a commercial package. S-Plus has a built-in graphical design intreface that some find convivial.

R has 2 major advantages for this course. Initially, you will find that it also has one inconvenience. However, this "inconvenience" will rapidly force you to acquire very good working habits. So, I see it as a third advantage.

The first advantage is that you can install it freely on you personal computer(s). This is important because it is by doing analyses that you will learn and eventually master biostatistics. This implies that you have easy and unlimited access to a statistical software package. The second advantage is that R can do everything in statistics. R was conceived to be extensible and has become the preferred tool for statisticians around the world. The question is not "Can R do this?" but rather "How can I do this in R?". And search engine are your friends.

No other software package offers you these two advantages.

The inconvenience of R is that one has to type commands (or copy and paste code) rather than use a menu and select options. If you do not know what command to use, nothing will happen. It is therefore not that easy when you start. However, it is possible to rapidly learn to make basic operations (open a data file, plot data, and run a simple analysis). And once you understand the operating principle, you can easily find examples on the Web for more complex analyses and graphs for which you can adapt the code.

This is exactly what you will do in the first lab to familiarize yourself with R.

Why is this inconvenience really an advantage in my mind? Because this way of doing things is more efficient and will save you time on the long run. I guarantee it. Believe me, you will never do an analysis only once. As you'll proceed through analyses, you will find data entry errors, discover that the analysis must be run separately for subgroups, find extra data, have to rerun the analysis on transformed data, or you will make some analytical error along the way. If you use a graphical interface with menus, redoing an analysis implies that you reclick here, enter values there, select some options, etc. Each of these steps is a potential source of error. If, instead, you use lines of codes, you only have to fix the code and submit to repeat instantaneously the entire analysis. And you can perfectly document what you did, leaving an audit trail for the future. This is how pros work and can document the quality of the results of their analyses.

Software installation

R

To install R on a computer, go to http://cran.r-project.org/. You will find compiled versions (binairies) for your preferred operating system (Windows, MacOS, Linux).

Note : R has already been installed on the lab computers (the version may be slightly different, but this should not matter).

Rstudio or VS code

RStudio and VS code are integrated development environment software or IDE. RStudio was develop specifically to work with R. VScode is more generela but work extremely well with R. Both are available on Windows, OS X and Linux

- RStudio: https://www.rstudio.com/products/rstudio/download/
- VScode: https://code.visualstudio.com/download

R libraries

R is essentially unlimited in terms of functions that can be used, because is relies on functions packages that can be added as extra components to use in R.

- Rmarkdown
- tinytex

Those 2 packages should be installed automatically with RStudio but I recommend to install them manually in case they are not. To do so, just copy-paste the text below in R terminal.

```
install.packages(c("rmarkdown", "tinytex"))
```

G*Power

G*Power est un programme gratuit, développé par des psychologues de l'Université de Dusseldorf en Allemagne. Le programme existe en version Mac et Windows. Il peut cependant être utilisé sous linux via Wine. G*Power vous permettra d'effectuer une analyse de puissance pour la majorité des tests que nous verrons au cours de la session sans avoir à effectuer des calculs complexes ou farfouiller dans des tableaux ou des figures décrivant des distributions ou des courbes de puissance.

Téléchargez le programme sur le site https://www.psychologie.hhu.de/arbeitsgruppen/allgemeine-psychologie-und-arbeitspsychologie/gpower.html

General laboratory instructions

- Bring a USB key or equivalent so you can save your work. Alternatively, email your results to yourself.
- Read the lab exercise before coming to the lab. Read the R code and come with questions about the code.
- During pre-labs, listen to the special instructions
- Do the laboratory exercises at your own rhythm, in teams. Then, I recommend that you start (complete?) the lab assignment so that you can benefit from the presence of the TA or prof.

- During your analyses, copy and paste results in a separate document, for example in your preferred word processing program. Annotate abundantly
- Each time you shut down R, save the history of your commands (ex: labo1.1 rHistory, labo1.2.rHistory, etc). You will be able to redo the lab rapidly, get code fragments, or more easily identify errors.
- Create your own "library" of code fragments (snippets). Annotate it abundantly. You will thank yourself later.

Notes about the manual

You will find explanations on the theory, R code and functions, IDE best practice and exercises with R.

The manual tries to highlight some part of the text using the following boxes and icons.

Exercises,

warnings,

warnings,

important points

notes

and tips

Resources {-}

This document was developped using the excellent bookdown de Yihui Xie. The manual is based on the previous lab manual *Findlay, Morin and Rundle, BIO4158 Laboratory manual for BIO4158.*

License

Figure 1: License Creative Commons

Chapitre 1

Introduction to R

After completing this laboratory exercise, you should be able to:

- Open R data files
- Import rectangular data sets
- Export R data to text files
- Verify that data were imported correctly
- Examine the distribution of a variable
- Examine visually and test for normality of a variable
- Calculate descriptive statistics for a variable
- Transform data

1.1 Packages and data needed for the lab

This labs needs the following:

- R packages:
  - ggplot2
- data files
  - ErablesGatineau.csv
  - sturgeon.csv

1.2 Importing and exporting data

There are multiple format to save data. The 2 most used formats with R are `.csv` and `.Rdata`.

- `.csv` files are used to store data in a simple format and are editable using any text editor (e.g. Word, Writer, atom, …) and spreadsheets (e.g. MS Excel, LO Calc). They can be read using the function `read.csv()` and created in R with `write.csv()`.
- `.Rdata` files are used to store not only data but any R object, however, those files can only be used in R. They are created using the `save()` function and read using the `load()` function.

Data for exercises and labs are provides in `.csv`.

1.2.1 Working directory

Potentially the most frequent error when starting with R is link to loading data or reading data from an external file in R.

A typical error message is:

```
Error in file(file, "rt") : cannot open the connection
In addition: Warning message:
In file(file, "rt") :
  cannot open file 'ou_est_mon_fichier.csv': No such file or directory
```

This type of error simply means that R cannot find the file you specified. By default, when R starts, a folder is defined as the base folder for R. This is the working directory. R by default will save any files in this folder and will start looking for files in this folder. So you need to specify to R where to look for files and where to save your files. This can be done in 3 different ways:

1. `file.choose()`. (not recommended, because not reproducible). This function will open a dialog box allowing you to click on the file you want. This is not recommended and can be long because you will have to do it absolutely every time you use R.
2. specify the complete path in the function. For example `read.csv("/home/julien/Documents/cours/BI`
   This is longer to type the first time and a bit tricky to get the correct path but after you can run the line of code and it works every time without trying to remember were you saved that damned file. However, this is specific to your own computer and would not work elsewhere.
3. specify a working directory with `setwd()`. This simplify tells R where to look for files and where to save files. (This is automatically done when using .Rmd files). Just set the working directory to where you want and after that all path will be relative to this working directory. The big advantage is that if you keep a similar folder structure for you R project it will be compatible and reproducible across all computer and OS

To know which folder is the workind directory simply type `getwd()`

When opening Rstudio by double-clicking on a file, it will automatically set the working irectory to the folder where this file is located. This can be super handy.

For all labs, I strongly recommend you to make a folder where you will save all your R scripts and data and use it as your working directory in R. For better organisation I suggest to save your data in a subfolder named `data All R code for data loading in the manual is based on that structure. This is why dat loading or saving code look like`data/my_file.xxx‘. If you follow it also all code for data loading can be simply copy-pasted and should work.

1.2.2 Opening a `.Rdata` file

You can double-click on the file and R/Rstudio should open. Alternatively, you can use `load()` function and specify the names (and path) of the file. For example to load the data

`ErablesGatineau.Rdata` in R which is located in the folder `data` in the working directory you can use:

```
load("data/ErablesGatineau.Rdata")
```

1.2.3 Open a `.csv` file

To import data saved in a `.csv` file, you need to use the `read.csv()` function. For example, to create a R object named `erables` which contain the data from the file `ErablesGatineau.csv`, you need to use:

```
erables <- read.csv("data/ErablesGatineau.csv")
```

Beware of the coma. If you are working in adifferent language (other than english), be careful because the decimal symbol might ot be the same. By default R use the point for the decimal sign. If the dat use the coma for the decimal then R would not be able to read the file correctly. In this case you can use `read.csv2()` or `read.data()` which should solve the problem.

To verify that the data were read and loaded properly, you can list all objects in memory with the `ls()` function, or get a more detailed description with `ls.str()`:

I do not recommend to use `ls.str()` since it can produce really long R ouputs when you have multiple R object loaded. I suggest instead to use the combination of `ls()` to get the list of all R objects and then `str()` only for the objects you want to look at.

```
ls()
```

```
## [1] "erables" "params"
```

```
str(erables)
```

```
## 'data.frame':    100 obs. of  3 variables:
##  $ station: chr  "A" "A" "A" "A" ...
##  $ diam   : num  22.4 36.1 44.4 24.6 17.7 ...
##  $ biom   : num  732 1171 673 1552 504 ...
```

R confirms that the object `erables`. `erables` is a data.frame that contains 100 observations (lines) of 3 variables (columns) : `station` , a variable of type Factor with 2 levels, and `diam` and `biom` that are 2 numeric variables.

1.2.4 Entering data in R

R is not the ideal environment to input data. It is possible, but the syntax is heavy and makes most people upset. Use your preferred worksheet program instead. It will be more efficient and less frustrating.

1.2.5 Cleaning up / correcting data

Another operation that can be frustrating in R. Our advice: unless you want to keep track of all corrections made (so that you can go back to the original data), do not change data in R. Return to the original data file (in a worksheet or database), correct the data there, and then reimport into R. It is simple to resubmit the few lines of code to reimport data. Doing things this way will leave you with a single version of your data file that has all corrections, and the code that allows you to repeat the analysis exactly.

1.2.6 Exporting data from R

You have 2 options: export data in `.csv` or in `.Rdata`

To export in `.Rdata` use the function `save()` to export in `.csv` use `write.csv()`

For example, to save teh object `mydata` in a file `wonderful_data.csv`that will be saved in your working directory you can type:

```
write.csv(mydata, file = "wonderful_data.csv", row.names = FALSE)
```

1.3 Preliminary examination of data

The first step of data analysis is to examine the data at hand. This examination will tell you if the data were correctly imported, whether the numbers are credible, whether all data came in, etc. This initial data examination often will allow you to detect unlikely observations, possibly due to errors at the data entry stage. Finally, the initial plotting of the data will allow you to visualize the major trends that will be confirmed later by your statistical analysis.

The file `sturgeon.csv` contains data on sturgeons from the Saskatchewan River. These data were collected to examine how sturgeon size varies among sexes (`sex`), sites (`location`), and years(`year`).

- Load the data from `sturgeon.csv` in a R object named `sturgeon`.
- use the function `str()` to check that the data was loaded and read correctly.

```
sturgeon <- read.csv("data/sturgeon.csv")
str(sturgeon)
```

```
## 'data.frame':    186 obs. of  9 variables:
##  $ fklngth : num  37 50.2 28.9 50.2 45.6 ...
##  $ totlngth: num  40.7 54.1 31.3 53.1 49.5 ...
```

```
##  $ drlngth : num  23.6 31.5 17.3 32.3 32.1 ...
##  $ rdwght  : num  15.95 NA 6.49 NA 29.92 ...
##  $ age     : int  11 24 7 23 20 23 20 7 23 19 ...
##  $ girth   : num  40.5 53.5 31 52.5 50 54.2 48 28.5 44 39 ...
##  $ sex     : chr  "MALE" "FEMALE" "MALE" "FEMALE" ...
##  $ location: chr  "THE_PAS" "THE_PAS" "THE_PAS" "THE_PAS" ...
##  $ year    : int  1978 1978 1978 1978 1978 1978 1978 1978 1978 1978 ...
```

1.3.1 Summary statistics

To get summary statistics on the contents of the data frame sturgeon, type the command:

```
summary(sturgeon)
```

```
##     fklngth         totlngth        drlngth          rdwght
##  Min.   :24.96   Min.   :28.15   Min.   :14.33   Min.   : 4.73
##  1st Qu.:41.00   1st Qu.:43.66   1st Qu.:25.00   1st Qu.:18.09
##  Median :44.06   Median :47.32   Median :27.00   Median :23.10
##  Mean   :44.15   Mean   :47.45   Mean   :27.29   Mean   :24.87
##  3rd Qu.:48.00   3rd Qu.:51.97   3rd Qu.:29.72   3rd Qu.:30.27
##  Max.   :66.85   Max.   :72.05   Max.   :41.93   Max.   :93.72
##                  NA's   :85      NA's   :13      NA's   :4
##       age             girth           sex              location
##  Min.   : 7.00   Min.   :11.50   Length:186         Length:186
##  1st Qu.:17.00   1st Qu.:40.00   Class :character   Class :character
##  Median :20.00   Median :44.00   Mode  :character   Mode  :character
##  Mean   :20.24   Mean   :44.33
##  3rd Qu.:23.50   3rd Qu.:48.80
##  Max.   :55.00   Max.   :73.70
##  NA's   :11      NA's   :85
##       year
##  Min.   :1978
##  1st Qu.:1979
##  Median :1979
##  Mean   :1979
##  3rd Qu.:1980
##  Max.   :1980
##
```

For each variable, R lists:

- the minimum
- the maximum
- the median that is the 50^{th} percentile, here the 93^{rd} value of the 186 observations ordered in ascending order
- values at the first (25%) and third quartile (75%)
- the number of missing values in the column.

Note that several variables have missing values (NA). Only the variables `fklngth` (fork length), `sex` , `location` , and `year` have 186 observations.

Beware of missing values

Several R functions are sensitive to missing values and you will frequently have to do your analyses on data subsets without missing data, or by using optional parameters in various commands. We will get back to this, but you should always pay attention and take note of missing data when you do analyses.

1.3.2 Histogram, empirical probability density, boxplot, and visual assessment of normality

Let's look more closely at the distribution of `fklngth`. The command `hist()` will create a histogram. For the histogram of `fklngth` in the `sturgeon` data frame, type the command:

```
hist(sturgeon$fklngth)
```

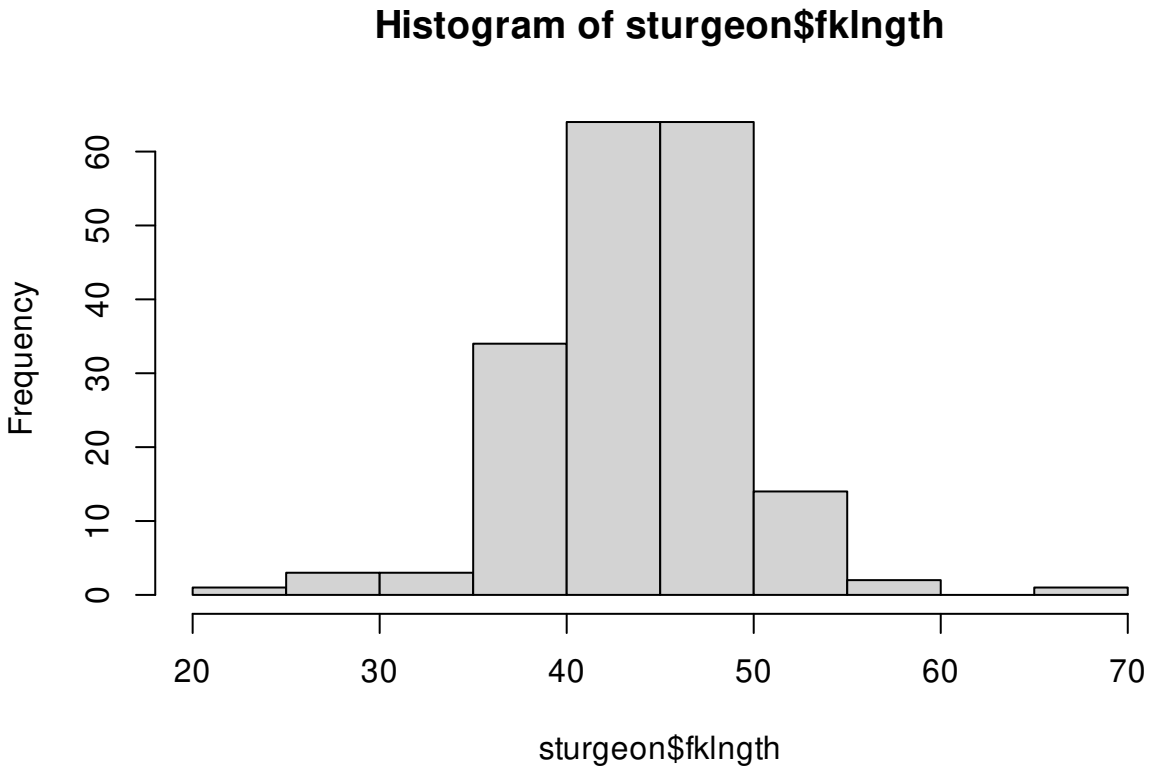


Figure 1.1: Histogram of fluke length of sturgeons

The data appear to be approximately normal. This is good to know.

Note that this syntax is a bit heavy as you need to prefix variable names by the data frame name `sturgeon$`. You can lighten the syntax by making the variables directly accessible by commands by typing the command `attach()`. However, I **strongly recommend not to use** it because it can lead to many problems hard to detect compare to the little benfit is provides

This histogram (Fig. **??**) is a very classical representation of the distribution. Histograms are not perfect however because their shape partly depends on the number of bins used, more so for small samples. One can do better, especially if you want to visually compare the observed distribution to a normal distribution. But you need to come up with a bit of extra R code based on the `ggplot2` .

```
## load ggplot2 if needed
library(ggplot2)

## use "sturgeon" dataframe to make plot called mygraph
# and define x axis as representing fklngth
mygraph <- ggplot(data = sturgeon, aes(x = fklngth))

## add data to the mygraph ggplot
mygraph <- mygraph +
  ## add semitransparent histogram
  geom_histogram(aes(y = ..density..),
    bins = 30, color = "black", alpha = 0.3
  ) +
  ##  add density smooth
  geom_density() +
  ## add observations positions or rug bars
  geom_rug() +
  ## add Gaussian curve adjusted to the data with mean and sd from fklngth
  stat_function(
    fun = dnorm,
    args = list(
      mean = mean(sturgeon$fklngth),
      sd = sd(sturgeon$fklngth)
    ),
    color = "red"
  )

## display graph
mygraph
```

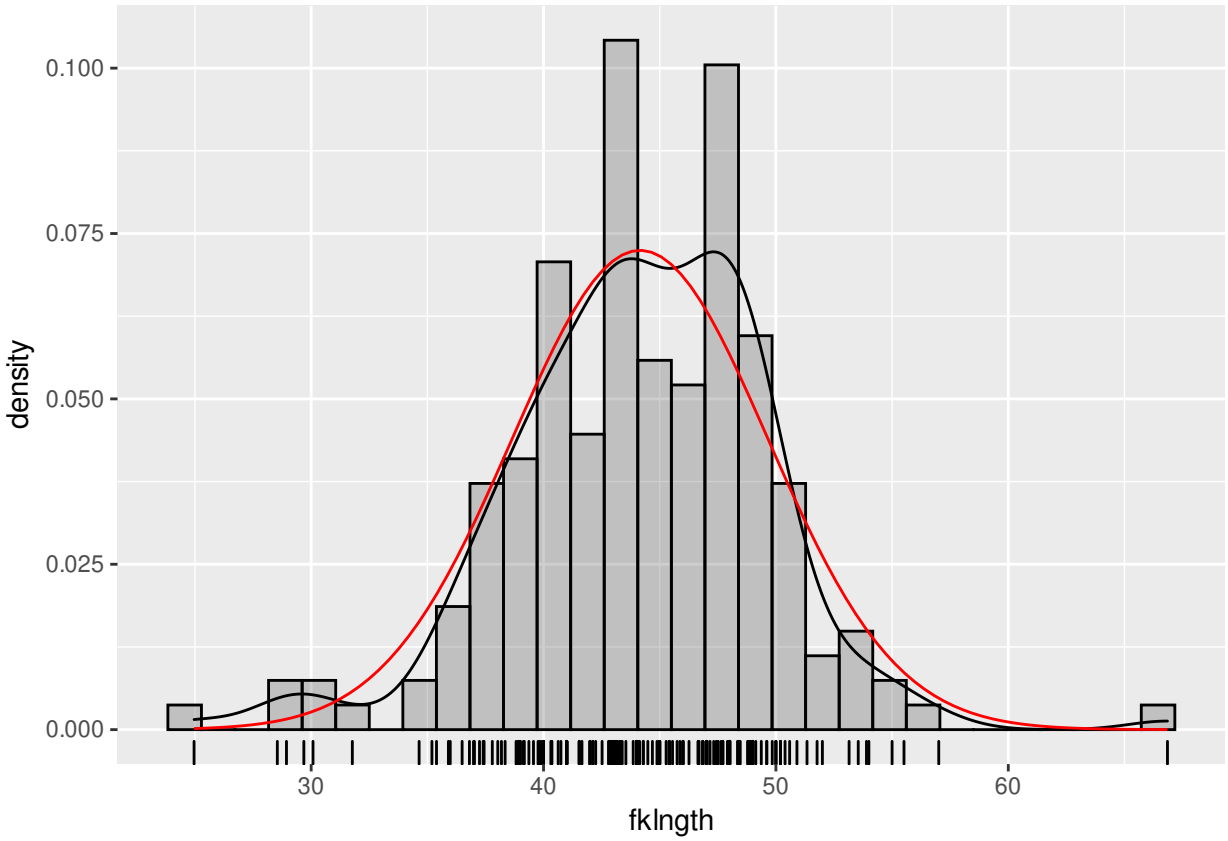
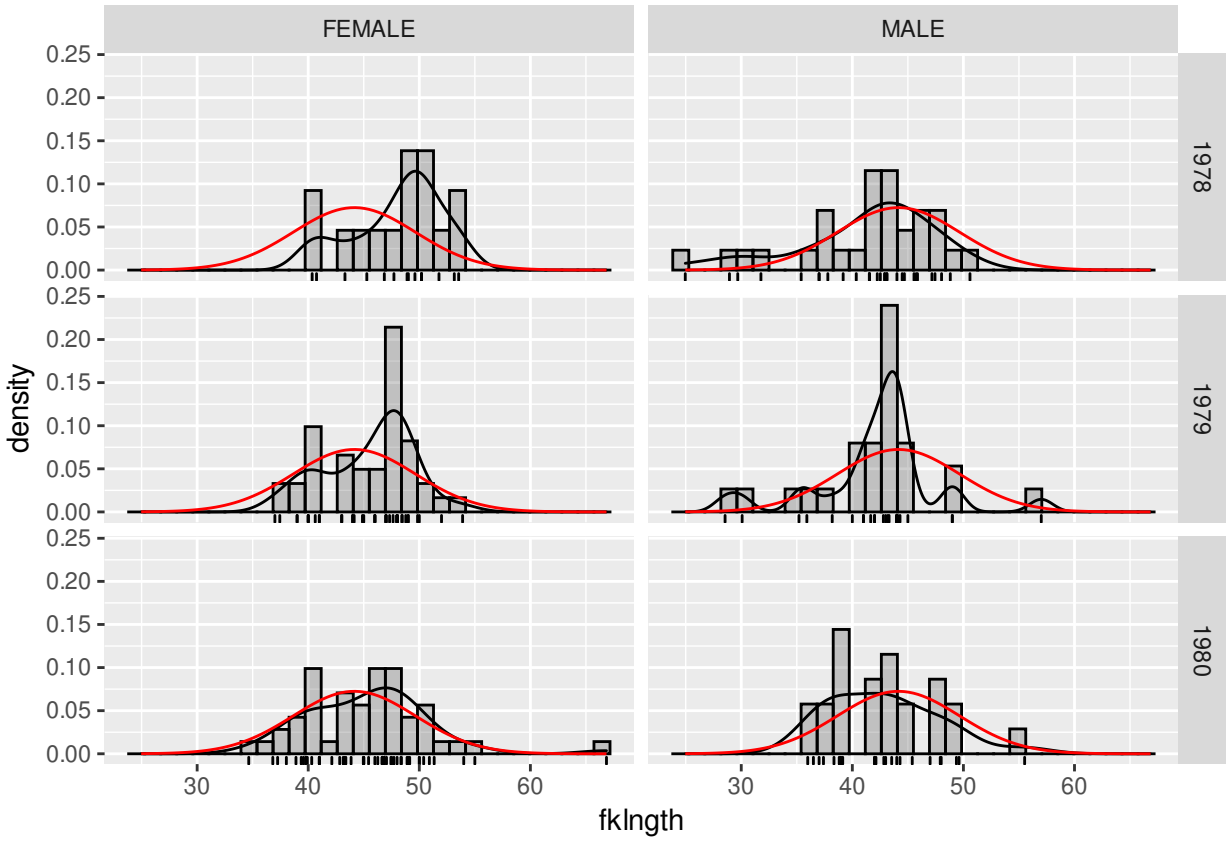


Figure 1.2: Distribution of fluke length in sturgeon plotted with ggplot

Each observation is represented by a short vertical bar below the x- axis (rug). The red line is the normal distribution with the same mean and standard deviation as the data. The other line is the empirical distribution, smoothed from the observations.

The ggplot object you just created (`mygraph`) can be further manipulated. For example, you can plot the distribution of `fklngth` per `sex` and `year` groups simply by adding a `facet_grid()` statement:

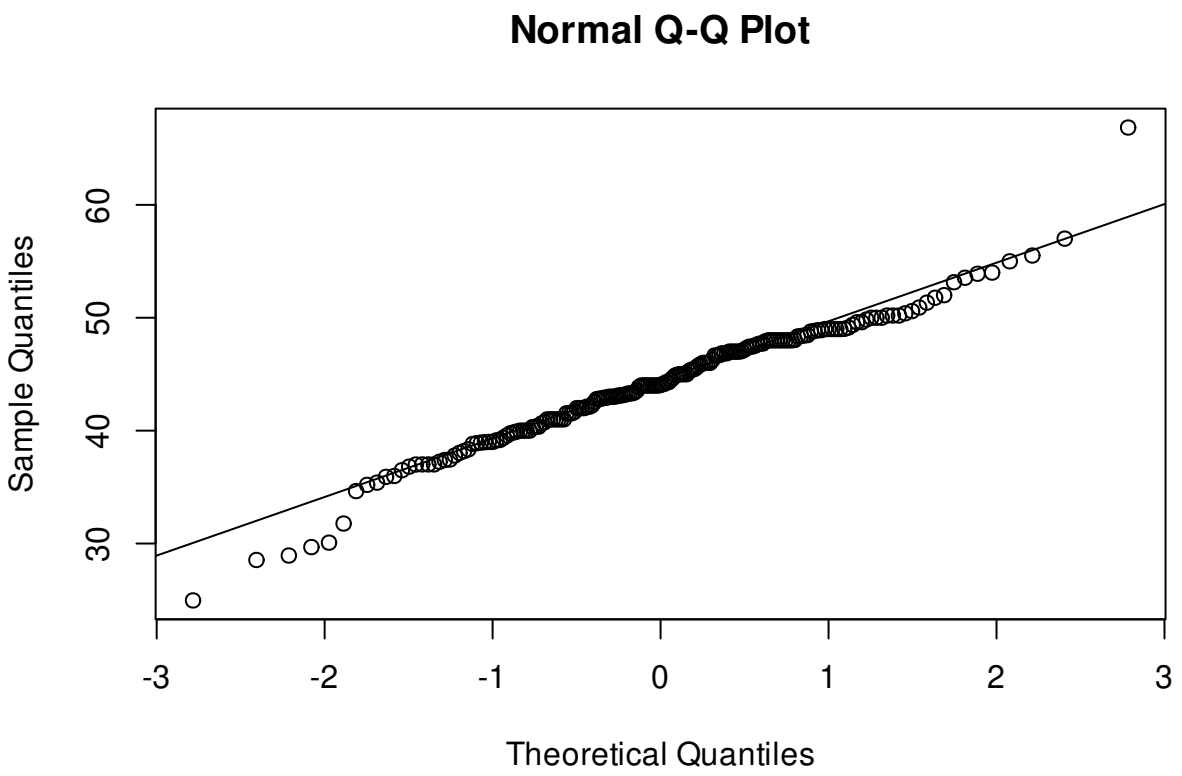
```
mygraph + facet_grid(year ~ sex)
```



Each panel contains the data distribution for one sex that year, and the recurring red curve is the normal distribution for the entire data set. It can serve as a reference to help visually evaluate differences among panels.

Another way to visually assess normality of data is the QQ plot that is obtained by the pair of commands `qqnorm()` and `qqline()`.

```
qqnorm(sturgeon$fklngth)
qqline(sturgeon$fklngth)
```



Perfectly normal data would follow the straight diagonal line. Here there are deviations in the tails of the distribution and a bit to the right of the center. Compare this representation to the two preceding graphs. You will probably agree that it is easier to visualize how data deviate from normality by looking at a histogram of an empirical probability density than by looking at the QQ plots. However, QQ plots are often automatically produced by various statistical routines and you should be able to interpret them. In addition, one can easily run a formal test of normality in R with the command `shapiro.test()` that computes a statistic (`W`) that measures how tightly data fall around the straight diagonal line of the QQ plot. If data fall perfectly on the line, then `W = 1`. If `W` is much less than 1, then data are not normal.

For the `fklngth` data:

```
shapiro.test(sturgeon$fklngth)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sturgeon$fklngth
## W = 0.97225, p-value = 0.0009285
```

W is close to 1, but far enough to indicate a statistically significant deviation from normality.

Visual examination of very large data sets is often made difficult by the superposition of data points. Boxplots are an interesting alternative. The command `boxplot(fklngth~sex, notch=TRUE)` produces a boxplot of `fklngth` for each `sex` , and adds whiskers.

```
boxplot(fklngth ~ sex, data = sturgeon, notch = TRUE)
```

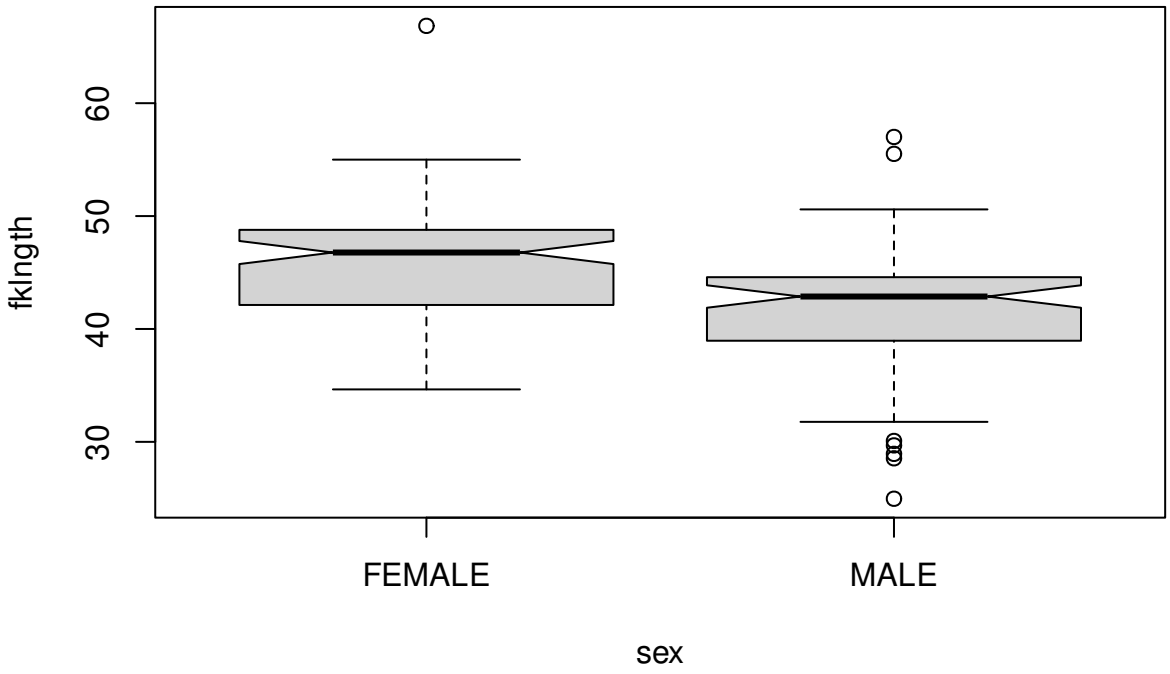


Figure 1.3: Boxplot of fluke length in strugeon by sex

The slightly thicker line inside the box of figure **??** indicates the median. The width of the notch is proportional to the uncertainty around the median estimate. One can visually assess the approximate statistical significance of differences among medians by looking at the overlap of the notches (here there is no overlap and one could tentatively conclude that the median female size is larger than the median male size). Boxes extend from the first to third quartile (the 25^{th} to 75^{th} percentile if you prefer). Bars (whiskers) extend above and below the boxes from the minimum to the maximum observed value or, if there are extreme values, from the smallest to the largest observed value within 1.5x the interquartile range from the median. Observations exceeding the limits of the whiskers (hence further away from the median than 1.5x the interquartile range, the range between the 25^{th} and 75^{th} percentile) are plotted as circles. These are outliers, possibly aberrant data.

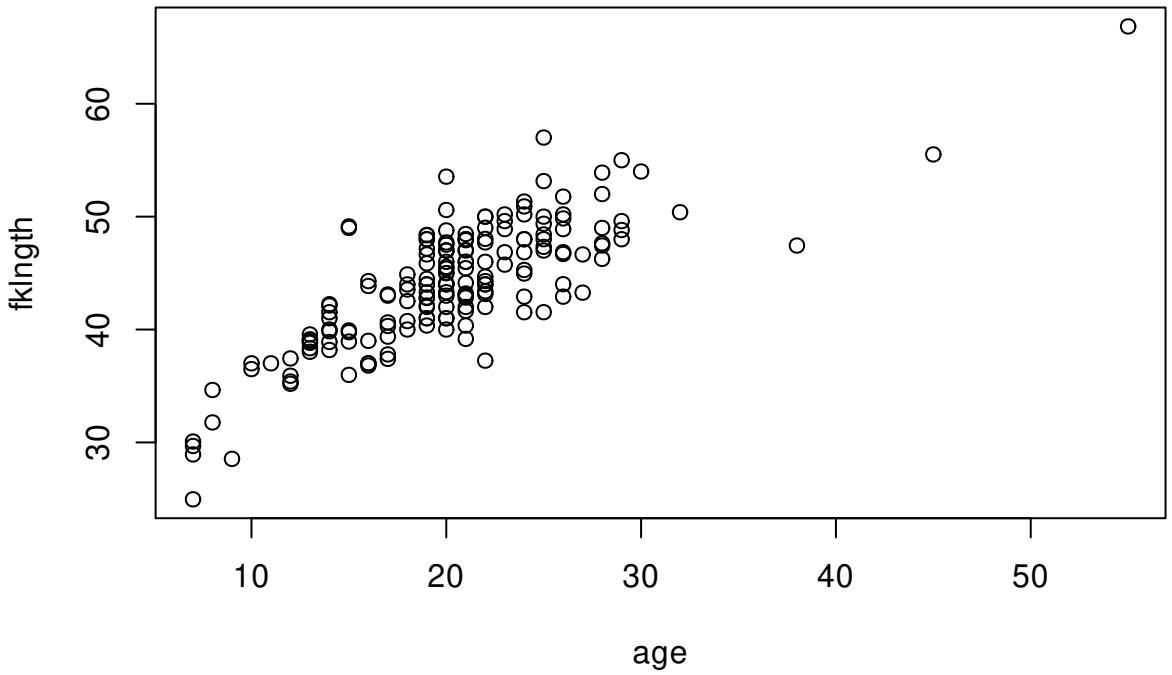
1.3.3 Scatterplots

In addition to histograms and other univariate plots, it is often informative to examine scatter plots. The command `plot(y~x)` produces a scatter plot of y on the vertical axis (the ordinate) vs x on the horizontal axis (abscissa).

Create a scatterplot of fklngth vs age using the plot() command.

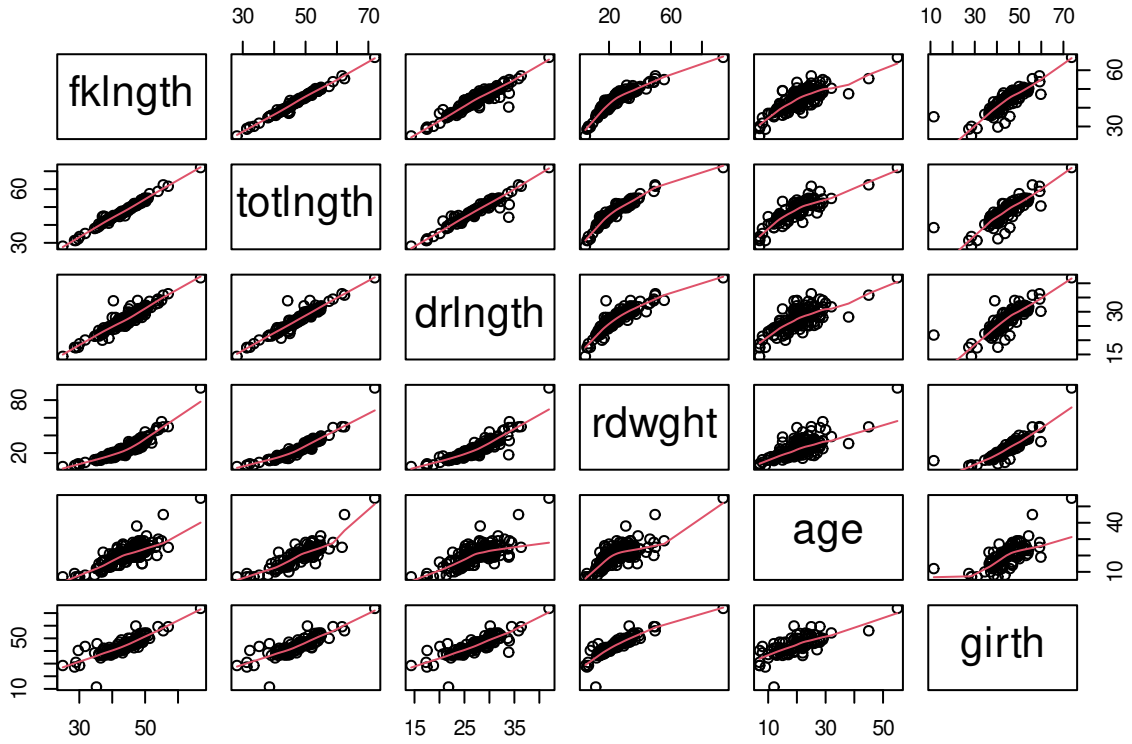
You should obtain:

```
plot(fklngth ~ age, data = sturgeon)
```



R has a function to create all pairwise scatterplots rapidly called `pairs()` . One of `pairs()` options is the addition of a lowess trace on each plot to that is a smoothed trend in the data. To get the plot matrix with the lowess smooth for all variables in the sturgeon data frame, execute the command `pairs(sturgeon, panel=panel.smooth)`. Howeber given the large number of variable in `sturgeon` we can limit the plot to the first 6 columns in the data.

```
pairs(sturgeon[, 1:6], panel = panel.smooth)
```



1.4 Creating data subsets

You will frequently want to do analyses on some subset of your data. The command `subset()` is what you need to isolate cases meeting some criteria. For example, to create a subset of the sturgeon data frame that contains only females caught in 1978, you could write:

```
sturgeon_female_1978 <- subset(sturgeon, sex == "FEMALE" & year == "1978")
sturgeon_female_1978
```

```
##      fklngth totlngth  drlngth rdwght age girth    sex   location year
## 2   50.19685 54.13386 31.49606     NA  24  53.5 FEMALE    THE_PAS 1978
## 4   50.19685 53.14961 32.28346     NA  23  52.5 FEMALE    THE_PAS 1978
## 6   49.60630 53.93701 31.10236  35.86  23  54.2 FEMALE    THE_PAS 1978
## 7   47.71654 51.37795 33.97638  33.88  20  48.0 FEMALE    THE_PAS 1978
## 15  48.89764 53.93701 29.92126  35.86  23  52.5 FEMALE    THE_PAS 1978
## 105 46.85039       NA 28.34646  23.90  24    NA FEMALE CUMBERLAND 1978
## 106 40.74803       NA 24.80315  17.50  18    NA FEMALE CUMBERLAND 1978
## 107 40.35433       NA 25.59055  20.90  21    NA FEMALE CUMBERLAND 1978
## 109 43.30709       NA 27.95276  24.10  19    NA FEMALE CUMBERLAND 1978
## 113 53.54331       NA 33.85827  48.90  20    NA FEMALE CUMBERLAND 1978
## 114 51.77165       NA 31.49606  35.30  26    NA FEMALE CUMBERLAND 1978
## 116 45.27559       NA 26.57480  23.70  24    NA FEMALE CUMBERLAND 1978
```

```
## 118 53.14961       NA 32.67717  45.30  25    NA FEMALE CUMBERLAND 1978
## 119 50.19685       NA 32.08661  33.90  26    NA FEMALE CUMBERLAND 1978
## 123 49.01575       NA 29.13386  37.50  22    NA FEMALE CUMBERLAND 1978
```

When using criteria to select cases, be careful of the `==` syntax to mean equal to. In this context, if you use a single `=`, you will not get what you want. The following table lists the most common criteria to create expressions and their R syntax.

	Operator	Explanation	Operator	Explanation
	==	Equal to	!=	Not equal to
	>	Larger than	<	Lower than
	>=	Larger than or equal to	<=	Lower than or equal to
	&	And (vectorized)	|	Or (vectorized)
	&&	And (control)	||	Or (control)
	!	Not		

Using the commands `subset()` and `hist()` , create a histogram for females caught in 1979 and 1980 (hint: `sex=="FEMALE" & (year =="1979" | year=="1980")`)

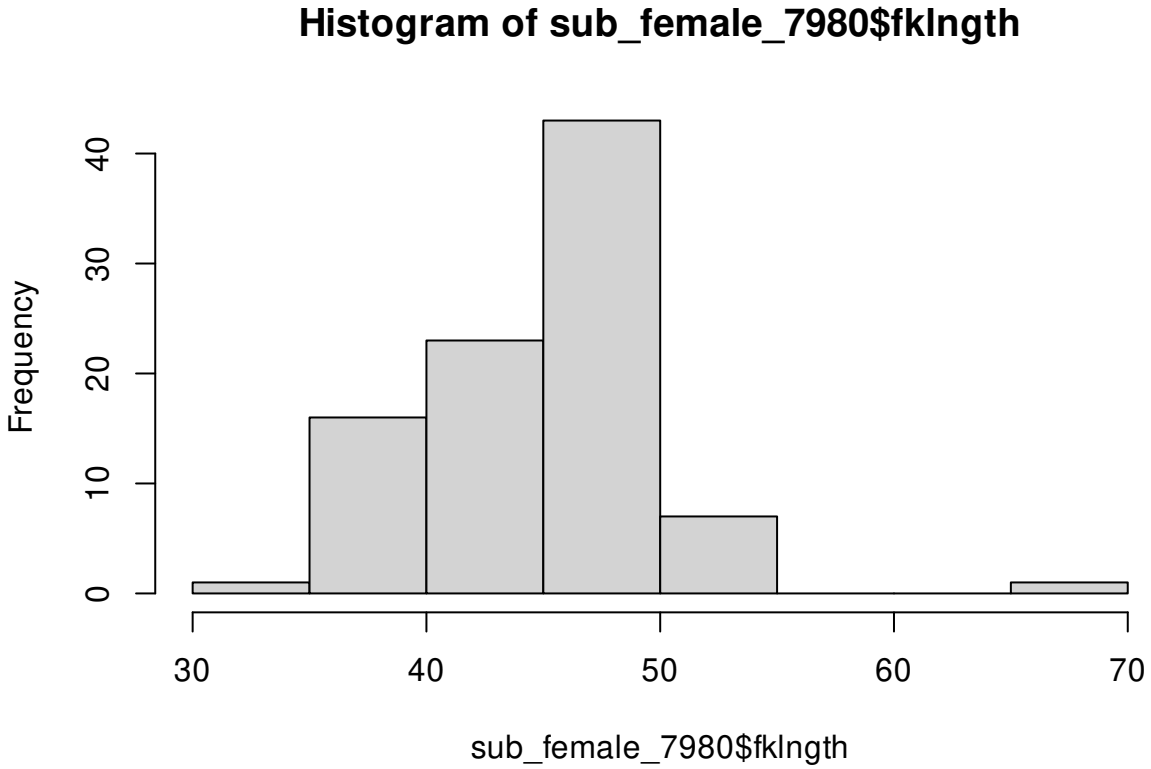


Figure 1.4: Distibution of fluke length of female sturgeons in 1979 and 1980

1.5 Data transformation

You will frequently transform raw data to better satisfy assumptions of statistical tests. R will allow you to do that easily. The most used functions are probably:

- `log()`
- `sqrt()`
- `ifelse()`

You can use these functions directly within commands, create vector variables, or add columns in data frames. To do a plot of the decimal log of fklngth vs age, you can simply use the `log10()` function within the plot command:

```
plot(log10(fklngth)~age, data = sturgeon)
```

To create a vector variable, an orphan variable if you wish, one that is not part of a data frame, called `lfklngth` and corresponding too the decimal log of `fklngth`, simply enter:

```
logfklngth <- log10(sturgeon$fklngth)
```

If you want this new variable to be added to a data frame, then you must prefix the variable name by the data frame name and the `$` symbol. For example to add the variable `lfkl` containing the decimal log of `fklngth` to the `sturgeon` data frame, enter:

```
sturgeon$lfkl <- log10(sturgeon$fklngth)
```

`lfkl` will be added to the data frame `sturgeon` for the R session. Do not forget to save the modified data frame if you want to keep the modified version. Or better, save you Rscript and do not forget to run the line of code again next time you need it.

For conditional transformations, you can use the function `ifelse()`. For example, to create a new variable called dummy with a value of 1 for males and 0 for females, you can use:

```
sturgeon$dummy <- ifelse(sturgeon$sex == "MALE", 1, 0)
```

1.6 Exercice

The file `salmonella.csv` contains numerical values for the variable called ratio for two environments (`milieu`: `IN VITRO` or `IN VIVO`) and for 3 strains (`souche`). Examine the ratio variable and make a graph to visually assess normality for the wild (SAUVAGE) strain.

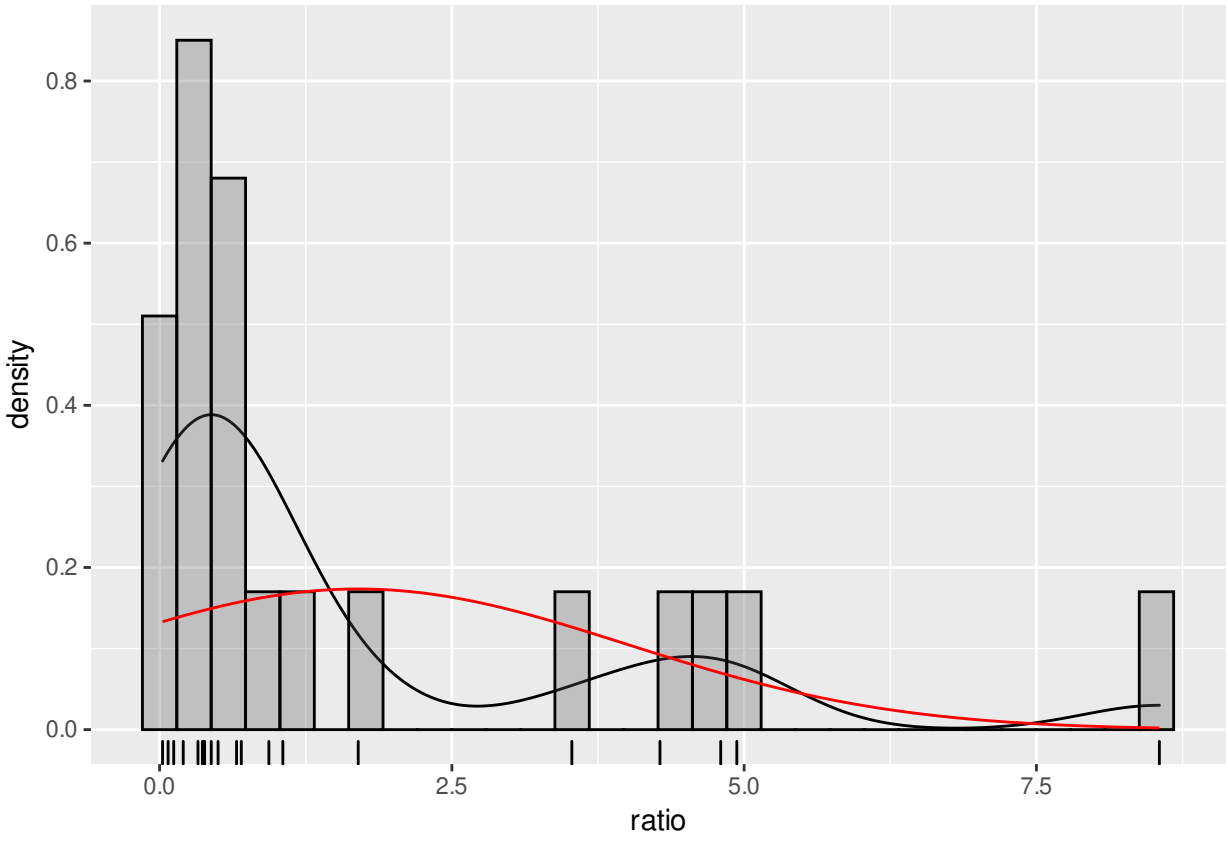


Figure 1.5: Distibution of infection ratios by the wild (SAUVAGE) strain of salmonella

Chapitre 2

Power Analysis with R and G*Power

After completing this laboratory, you should :

- be able to compute the power of a t-test with G*Power and R
- be able to calculate the required sample size to achieve a desired power level with a t-test
- be able to calculate the detectable effect size by a t-test given the sample size, the power and α
- understand how power changes when sample size increases, the effect size changes, or when α decreases
- understand how power is affected when you change from a two-tailed to a one-tailed test.

2.1 The theory

2.1.1 What is power?

Power is the probability of rejecting the null hypothesis when it is false

2.1.2 Why do a power analysis?

Assess the strength of evidence

Power analysis, performed after accepting a null hypothesis, can help assess the probability of rejecting the null if it were false, and if the magnitude of the effect was equal to that observed (or to any other given magnitude). This type of *a posteriori* analysis is very common.

Design better experiments

Power analysis, performed prior to conducting an experiment (but most often after a preliminary experiment), can be used to determine the number of observations required to detect an effect of a given magnitude with some probability (the power). This type of *a priori* experiment should be more common.

Estimate minimum detectable effect

Sampling effort is often predetermined (when you are handed data of an experiment already completed), or extremely constrained (when logistics dictates what can be done). Whether it is *a priori* or *a posteriori*, power analysis can help you estimate, for a fixed sample size and a given power, what is the minimum effect size that can be detected.

2.1.3 Factors affecting power

For a given statistical test, there are 3 factors that affect power.

Decision criteria

Power is related to α, the probability level at which one rejects the null hypothesis. If this decision criteria is made very strict (i.e. if critical α is set to a very low value, like 0.1% or $p = 0.001$), then power will be lower than if the critical α was less strict.

Sample size

The larger the sample size, the larger the power. As sample size increases, one's ability to detect small effect sizes as being statistically significant gets better.

Effect size

The larger the effect size, the larger the power. For a given sample size, the ability to detect an effect as being significant is higher for large effects than for small ones. Effect size measures how false the null hypothesis is.

2.2 What is G*Power?

G*Power is free software developed by quantitative psychologists from the University of Dusseldorf in Germany. It is available in MacOS and Windows versions. It can be run under Linux using Wine or a virtual machine.

G*Power will allow you to do power analyses for the majority of statistical tests we will cover during the term without making lengthy calculations and looking up long tables and figures of power curves. It is a really useful tool that you need to master.

It is possible to perform all analysis made by G*Power in R, but it requires a bit more code, and a better understanding of the process since everything should be coded by hand. In simple cases, R code is also provided.

Download the software **here** and install it on your computer and your workstation (if it is not there already).

2.3 How to use G*Power

2.3.1 General Principle

Using G*Power generally involves 3 steps:

1. Choosing the appropriate test
2. Choosing one of the 5 types of available power analyses
3. Enter parameter values and press the **Calculate** button

2.3.2 Types of power analyses

First, α is define as the probability level at which one rejects the null hypothesis, and β is $1-power$.

A priori

Computes the sample size required given β, α, and the effect size. This type of analysis is useful when planning experiments.

Compromise

Computes α and β for a given α/β ratio, sample size, and effect size. Less commonly used (I have never used it myself) although it can be useful when the α/β ratio has meaning, for example when the cost of type I and type II errors can be quantified.

Criterion

Computes α for a given β, sample size, and effect size. In practice, I see little interest in this. Let me know if you see something I don't!

Post-hoc

Computes the power for a given α, effect size, and sample size. Used frequently to help in the interpretation of a test that is not statistically significant, but only if an effect size that is biologically significant is used (and not the observed effect size). Not relevant when the test is significant. Sensitivity. Computes the detectable effect size for a given β ,α, and sample size. Very useful at the planning stage of an experiment.

2.3.3 How to calculate effect size

G*Power can perform power analyses for several statistical tests. The metric for effect size depends on the test. Note that other software packages often use different effect size metrics and that it is important to use the correct one for each package. G*Power has an effect size calculator for many tests that only requires you to enter the relevant values. The following table lists the effect size metrics used by G*Power for the various tests.

Test	Taille d'effet	Formule
t-test on means	d	$d = \frac{\lvert\mu_1 - \mu_2\rvert}{\sqrt{({s_1}^2 + {s_2}^2)/2}}$
t-test on correlations	r	
other t-tests	f	$f = \frac{\mu_1}{\sigma}$
F-test (ANOVA)	f	$f = \frac{\sqrt{\frac{\sum_{i=1}^{k}(\mu_i - \mu)^2}{k}}}{\sigma}$
other F-tests	f^2	$f^2 = \frac{{R_p}^2}{1 - {R_p}^2}$ R_p is the squared partial correlation coefficient
Chi-square test	w	$w = \sqrt{\sum_{i=1}^{m} \frac{(p_{0i} - p_{1i})^2}{p_{0i}}}$ p_{0i} and p_{1i} are theporportion in category i predicted by the null, $_0$, and alternative, $_1$, hypothesis

2.4 Power analysis for a t-test on two independent means

The objective of this lab is to learn to use G*Power and understand how the 4 parameters of power analyses (α, β, sample size and effect size) are related to each other. For this, you will only use the standard t-test to compare two independent means. This is the test most used by biologists, you have all used it, and it will serve admirably for this lab. What you will learn today will be applicable to all other power analyses.

Jaynie Stephenson studied the productivity of streams in the Ottawa region. She has measured fish biomass in 36 streams, 18 on the Shield and 18 in the Ottawa Valley. She found that fish biomass was lower in streams from the valley (2.64 g/m^2 , standard deviation = 3.28) than from the Shield (3.31 g/m^2 , standard deviation = 2.79.).

When she tested the null hypothesis that fish biomass is the same in the two regions by a t-test, she obtained:

```
Pooled-Variance Two-Sample t-Test
t = -0.5746, df = 34, p-value = 0.5693
```

She therefore accepted the null hypothesis (since p is much larger than 0.05) and concluded that fish biomass is the same in the two regions.

2.4.1 Post-hoc analysis

Using the observed means and standard deviations, we can use G*Power to calculate the power of the two-tailed t-test for two independent means, using the observed effect size (the difference

between the two means, weighted by the standard deviations) for $\alpha = 0.05$.

Start G*Power.

1. In ***Test family** , choose: t tests
2. For **Statistical test** , choose: Means: Difference between two independent means (two groups)
3. For **Type of power analysis** , choose: Post hoc: Compute achieved power - given α, sample size, and effect size
4. At **Input Parameters** ,

- in the box **Tail(s)** , chose: Two,
- check that α **err prob** is equal to 0.05
- Enter 18 for the **Sample size** of group 1 and of group 2
- then, to calculate effect size (d), click on **Determine =>**

5. In the window that opens,

- select **n1 = n2** , then
- enter the two means (**Mean group** 1 et 2)
- the two standard deviations(**SD group** 1 et 2)
- click on **Calculate** and **transfer to main window**

6. After you click on the **Calculate button** in the main window, you should get the following:

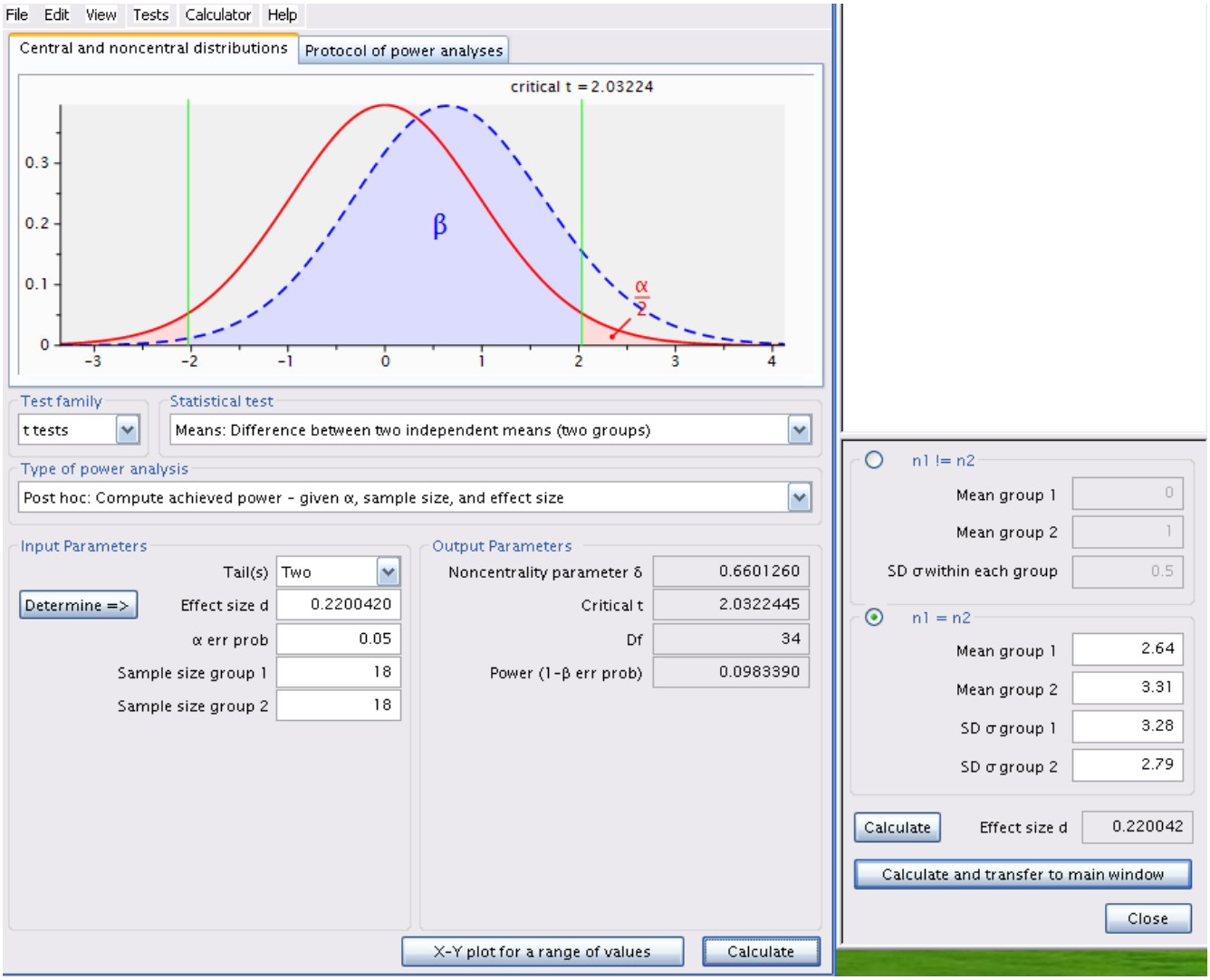


Figure 2.1: Post-hoc analysis with estimated effect size

Similar analysis can be done in R. You first need to calculate the effect size `d` for a t-test comparing 2 means, and then use the `pwr.t.test()` function from the `pwr` . The easiest is to create anew function in R to estimate the effect size `d`since we are going to reuse it multiple times during the lab.

```r
# load package pwr
library(pwr)
# define d for a 2 sample t-test
d <- function(u1, u2, sd1, sd2) {
  abs(u1 - u2) / sqrt((sd1^2 + sd2^2) / 2)
}

# power analysis
pwr.t.test(
```

```
  n = 18,
  d = d(u1 = 2.64, sd1 = 3.28, u2 = 3.31, sd2 = 2.79),
  sig.level = 0.05,
  type = "two.sample")
```

```
##
##      Two-sample t test power calculation
##
##               n = 18
##               d = 0.220042
##       sig.level = 0.05
##           power = 0.09833902
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

```
# plot similar to G*Power
x <- seq(-4, 4, length = 200)
plot(x, dnorm(x), type = "l", col = "red", lwd = 2)
qc <- qt(0.025, 16)
abline(v = qc, col = "green")
abline(v = -qc, col = "green")
lines(x, dnorm(x, mean = (3.31 - 2.64)), type = "l", col = "blue", lwd = 2)

# power corresponds to the shaded area
y <- dnorm(x, mean = (3.31 - 2.64))
polygon(
  c(x[x <= -qc], -qc), c(y[x <= -qc], 0),
  col = rgb(red = 0, green = 0.2, blue = 1, alpha = 0.5))
```

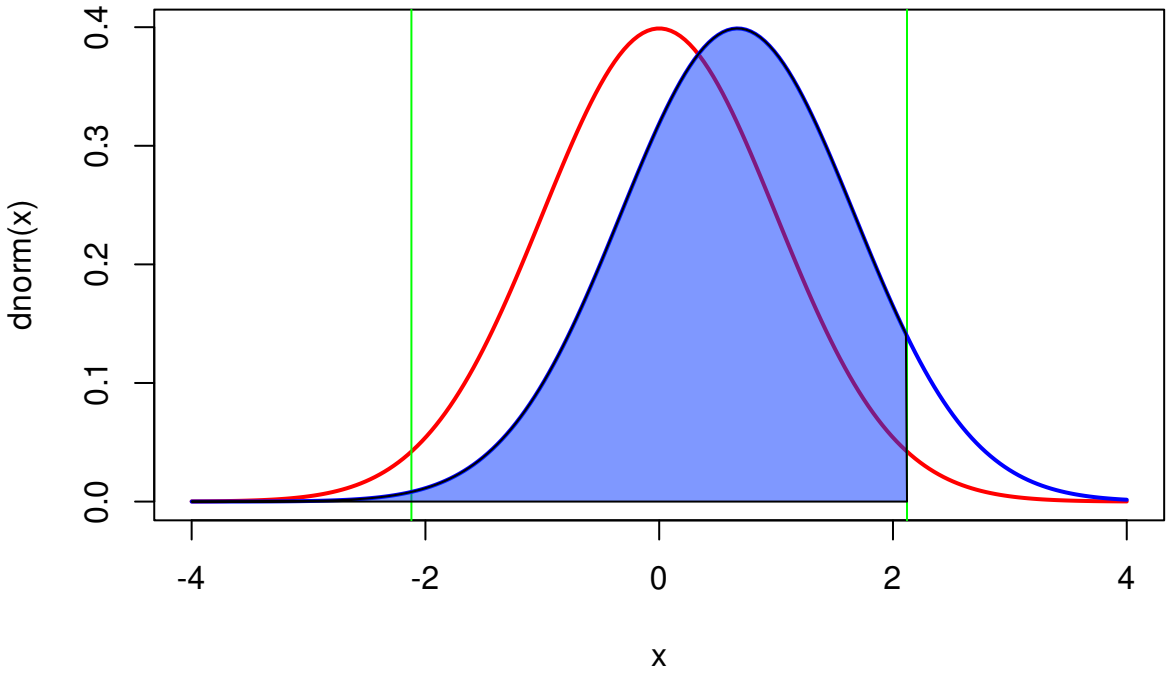


Figure 2.2: Post-hoc analysis with estimated effect size in R

Let's examine the figure **??**.

- The curve on the left, in red, corresponds to the expected distri- bution of the t-statistics when H_0 is true (*i.e.* when the two means are equal) given the sample size (18 per region) and the observed standard deviations.
- The vertical green lines correspond to the critical values of t for $\alpha = 0.05$ and a total sample size of 36 (2x18).
- The shaded pink regions correspond to the rejection zones for H_0. If Jaynie had obtained a *t-value* outside the interval delimited by the critical values ranging from -2.03224 to 2.03224, she would then have rejected H_0 , the null hypothesis of equal means. In fact, she obtained a t-value of -0.5746 and concluded that the biomass is equal in the two regions.
- The curve on the right, in blue, corresponds to the expected dis- tribution of the t-statistics if H_1 is true (here H_1 is that there is a difference in biomass between the two regions equal to $3.33 - 2.64 = 0.69 g/m^2$, given the observed standard deviations). This distribution is what we should observe if H_1 was true and we repeated a large number of times the experiment using random samples of 18 streams in each of the two regions and calculated a t-statistic for each sample. On average, we would obtain a t-statistic of about 0.6.
- Note that there is considerable overlap of the two distributions and that a large fraction of the surface under the right curve is within the interval where H_0 is accepted between the two vertical green lines at -2.03224 and 2.03224. This proportion, shaded in blue under the distribution on the right is labeled β and corresponds to the risk of *type II error* (accept H_0

when H_1 is true).

- Power is simply $1 - \beta$, and is here 0.098339. Therefore, if the mean biomass differed by $0.69 g/m^2$ between the two regions, Jaynie had only 9.8% chance of being able to detect it as a statistically significant difference at =5% with a sample size of 18 streams in each region.

Let's recapitulate: The difference in biomass between regions is not statistically significant according to the t-test. It is because the difference is relatively small relative to the precision of the measurements. It is therefore not surprising that that power, i.e. the probability of detecting a statistically significant difference, is small. Therefore, this analysis is not very informative.

Indeed, a post hoc power analysis using the observed effect size is not useful. It is much more informative to conduct a post hoc power analysis for an effect size that is different from the observed effect size. But what effect size to use? It is the biology of the system under study that will guide you. For example, with respect to fish biomass in streams, one could argue that a two fold change in biomass (say from 2.64 to 5.28 g/m^2) has ecologically significant repercussions. We would therefore want to know if Jaynie had a good chance of detecting a difference as large as this before accepting her conclusion that the biomass is the same in the two regions. So, what were the odds that Jaynie could detect a difference of 2.64 g/m^2 between the two regions? G*Power can tell you if you cajole it the right way.

Change the mean of group 2 to 5.28, recalculate effect size, and click on Calculate to obtain figure **??**.

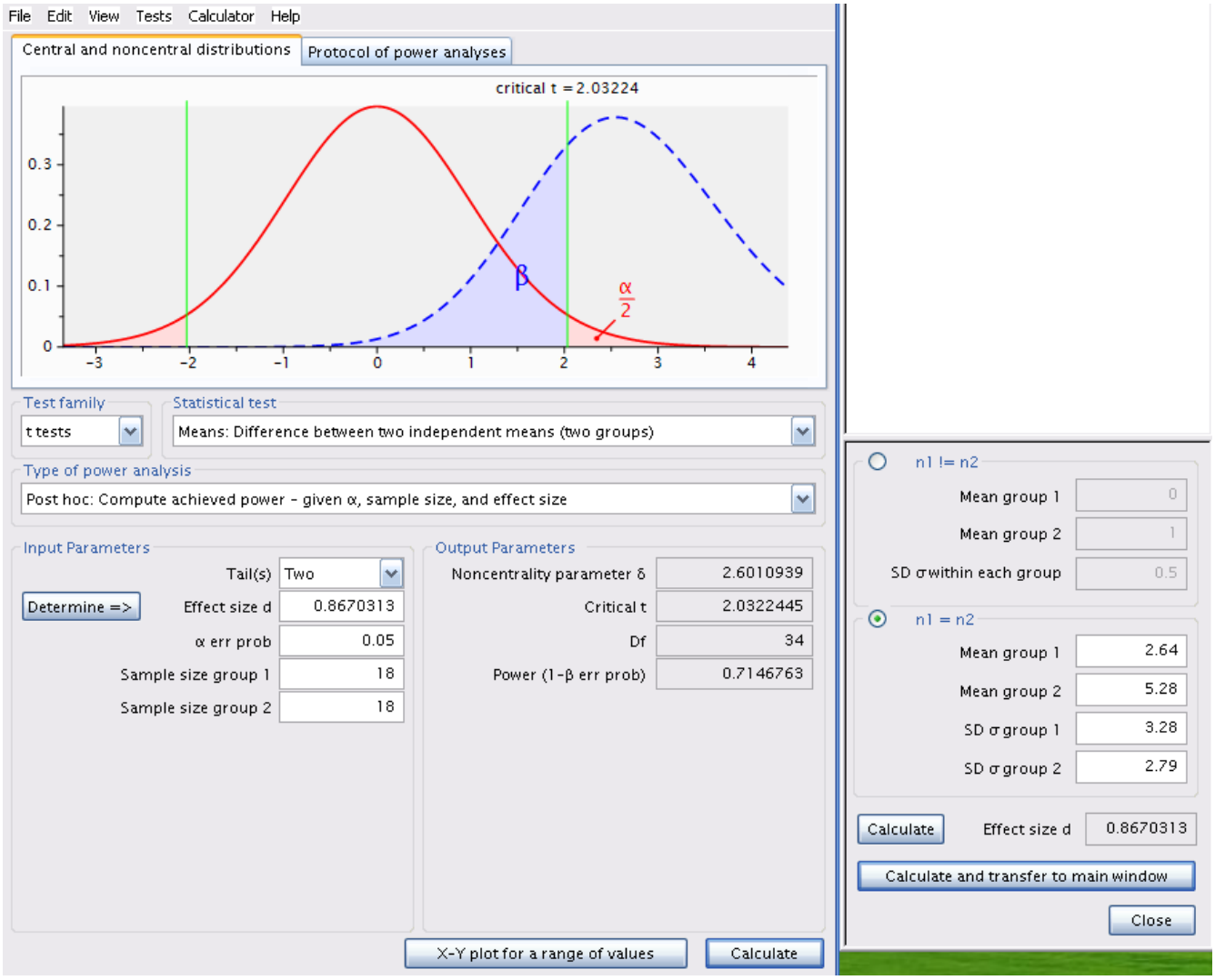


Figure 2.3: Post-hoc analysis using an effect size different from the one estimated

Same analysis using R (without all the code for the interesting but not really useful plot)

```
pwr.t.test(
  n = 18,
  d = d(u1 = 2.64, sd1 = 3.28, u2 = 5.28, sd2 = 2.79),
  sig.level = 0.05,
  type = "two.sample")
```

```
##
##      Two-sample t test power calculation
##
##               n = 18
##               d = 0.8670313
##       sig.level = 0.05
```

```
##           power = 0.7146763
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

The power is 0.71, therefore Jaynie had a reasonable chance (71%) of detecting a doubling of biomass with 18 streams in each region.

Note that this post hoc power analysis, done for an effect size considered biologically meaningful, is much more informative than the preceeding one done with the observed effect size (which is what too many students do because it is the default of so many power calculation programs). Jaynie did not detect a difference between the two regions. There are two possibilities: 1) there is really no difference between the regions, or 2) the precision of measurements is so low (because the sample size is small and/or there is large variability within a region) that it is very unlikely to be able to detect even large differences. The second power analysis can eliminate this second possibility because Jaynie had 71% chances of detecting a doubling of biomass.

2.4.2 A priori analysis

Suppose that a difference in biomass of $3.31 - 2.64 = 0.67 g/m^2$ can be ecologically significant. The next field season should be planned so that Jaynie would have a good chance of detecting such a difference in fish biomass between regions. How many streams should Jaynie sample in each region to have 80% of detecting such a difference (given the observed variability)?

Change the type of power analysis in G*Power to **A priori: Compute sample size - given α , power, and effect size**. Ensure that the values for means and standard deviations are those obtained by Jaynie. Recalculate the effect size metric and enter 0.8 for power and you will obtain (**??**.

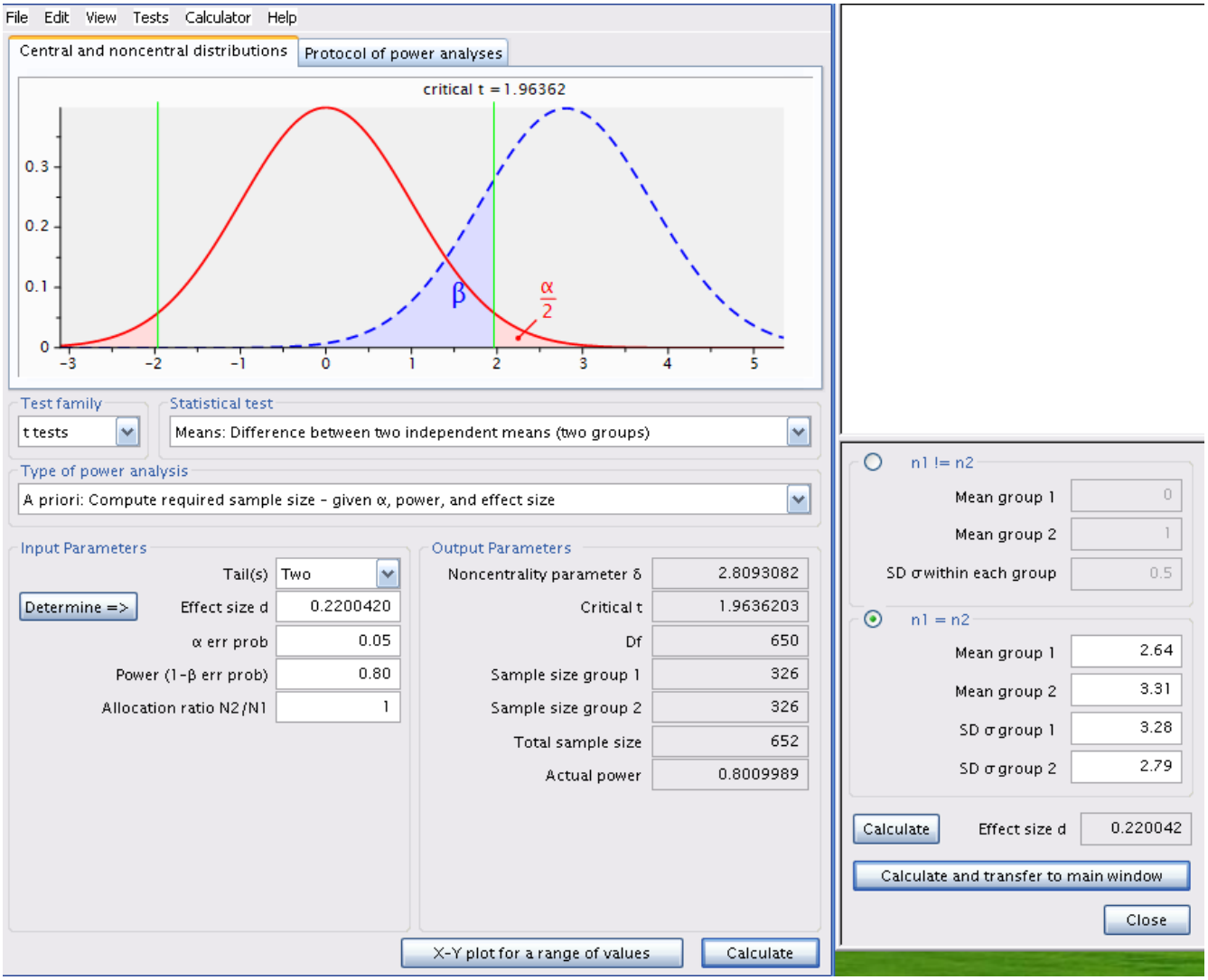


Figure 2.4: A priori analysis

```
pwr.t.test(
  power = 0.8,
  d = d(u1 = 2.64, sd1 = 3.28, u2 = 3.31, sd2 = 2.79),
  sig.level = 0.05,
  type = "two.sample")
```

```
##
##      Two-sample t test power calculation
##
##               n = 325.1723
##               d = 0.220042
##       sig.level = 0.05
##           power = 0.8
##     alternative = two.sided
```

```
##
## NOTE: n is number in *each* group
```

Ouch! The required sample would be of 326 streams in each region! It would cost a fortune and require several field teams otherwise only a few dozen streams could be sampled over the summer and it would be very unlikely that such a small difference in biomass could be detected. Sampling fewer streams would probably be in vain and could be considered as a waste of effort and time: why do the work on several dozens of streams if the odds of success are that low?

If we recalculate for a power of 95%, we find that 538 streams would be required from each region. Increasing power means more work!

```
pwr.t.test(
  power = 0.95,
  d = d(u1 = 2.64, sd1 = 3.28, u2 = 3.31, sd2 = 2.79),
  sig.level = 0.05,
  type = "two.sample")
```

```
##
##      Two-sample t test power calculation
##
##               n = 537.7286
##               d = 0.220042
##       sig.level = 0.05
##           power = 0.95
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

2.4.3 Sensitivity analysis - Calculate the detectable effect size

Given the observed variability, a sampling effort of 18 streams per region, and with $\alpha = 0.05$, what effect size could Jaynie detect with 80% probability $\beta = 0.2$)?

Change analysis type in G*Power to **Sensitivity: Compute required effect size - given α , power, and sample size** and size is 18 in each region.

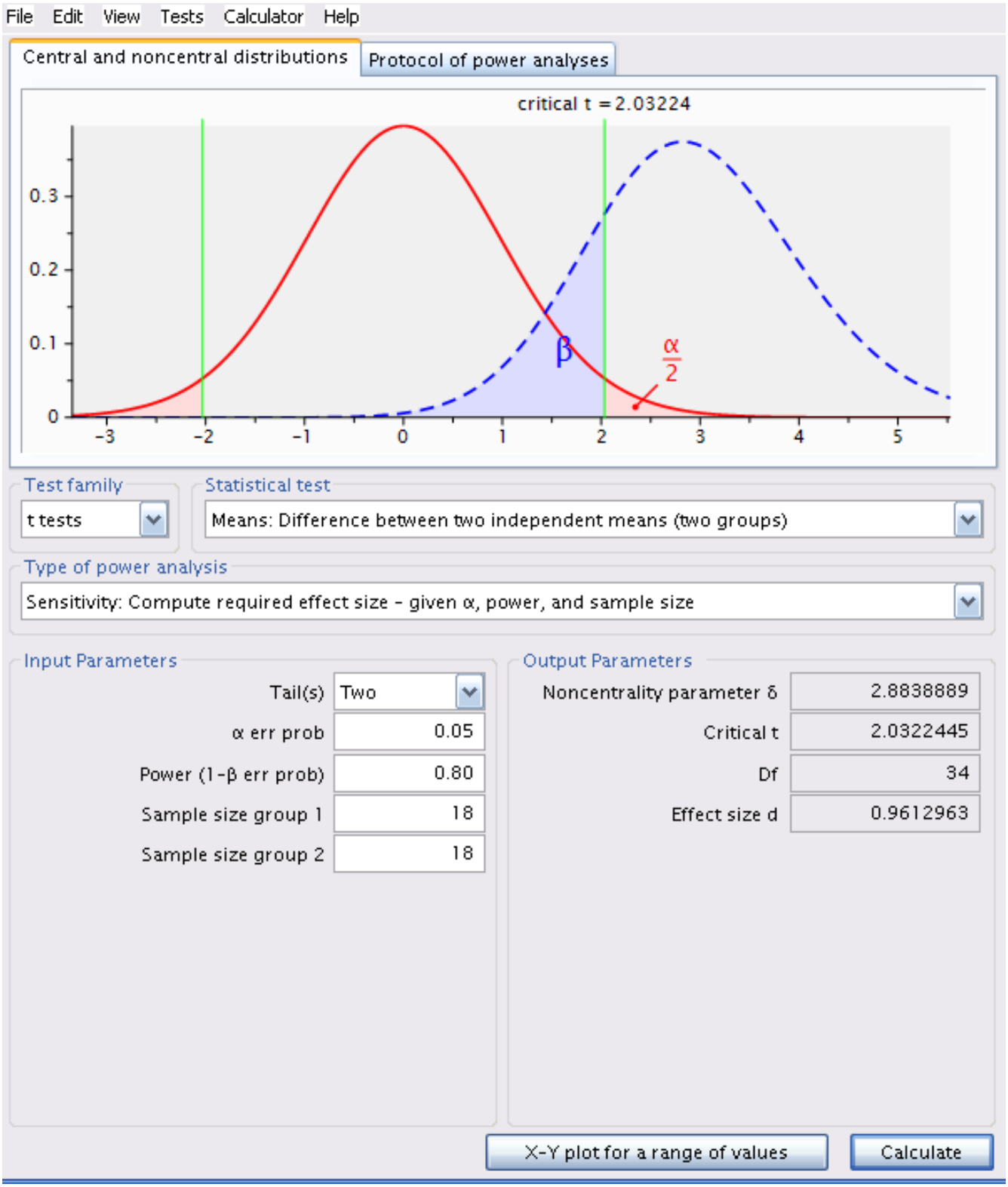


Figure 2.5: Analyse de sensitivité

```
pwr.t.test(
  power = 0.8,
  n = 18,
  sig.level = 0.05,
  type = "two.sample")
```

```
##
##      Two-sample t test power calculation
##
##               n = 18
##               d = 0.9612854
##       sig.level = 0.05
##           power = 0.8
##     alternative = two.sided
##
## NOTE: n is number in *each* group
```

The detectable effect size for this sample size, $\alpha = 0.05$ and $\beta = 0.2$ (or power of 80%) is 0.961296.

Attention, this effect size is the metric `d` and is dependent on sampling variability.

Here, `d` is approximately equal to

$$d = \frac{|\bar{X}_1 \bar{X}_2|}{\sqrt{\frac{{s_1}^2 + {s_2}^2}{2}}}$$

To convert this d value without units into a value for the detectable difference in biomass between the two regions, you need to multiply `d` by the denominator of the equation.

$$|\bar{X}_1 - \bar{X}_2| = d * \sqrt{\frac{{s_1}^2 + {s_2}^2}{2}}$$

In R this can be done with the following code

```
pwr.t.test(
  power = 0.8,
  n = 18,
  sig.level = 0.05,
  type = "two.sample")$d * sqrt((3.28^2 + 2.79^2) / 2)
```

```
## [1] 2.926992
```

Therefore, with 18 streams per region, $\alpha = 0.05$ and $\beta = 0.2$ (so power of 80%), Jaynie could detect a difference of 2.93 g/m^2 between regions, a bit more than a doubling of biomass.

2.5 Important points to remember

- Post hoc power analyses are relevant only when the null hypothesis is accepted because it is impossible to make a *type II error* when rejecting H_0 .
- With very large samples, power is very high and minute differences can be statistically detected, even if they are not biologically significant.
- When using a stricter significance criteria ($\alpha < 0.05$) power is reduced.
- Maximizing power implies more sampling effort, unless you use a more liberal statistical criteria ($\alpha > 0.05$)
- The choice of β is somewhat arbitrary. $\beta = 0.2$ (power of 80%) is considered relatively high by most.

Chapitre 3

Correlation and simple linear regression

After completing this laboratory exercise, you should be able to:

- Use R to produce a scatter plot of the relationship between two variables.
- Use R to carry out some simple data transformations.
- Use R to compute the Pearson product-moment correlation between two variables and assess its statistical significance.
- Use R to compute the correlation between pairs of ranked vari- ables using the Spearman rank correlation and Kendall's tau.
- Use R to assess the significance of pairwise comparisons from a generalized correlation matrix using Bonferroni-adjusted proba- bilities.
- Use R do a simple linear regression
- Use R to test the validity of the assumptions underlying simple lin- ear regression
- Use R to assess significance of a regression by the bootstrap method
- Quantify effect size in simple regression and perform a power analysis using G*Power

3.1 R packages and data

For this la b you need:

- R packages:
  - car
  - lmtest
  - boot
  - pwr
  - ggplot
  - performance
- data:
  - sturgeon.csv

You need to load the packages in R with `library()` and if need needed install them first with `install.packages()` For the data, load them using the `read.csv()` function.

```
library(car)
library(lmtest)
library(performance)
library(boot)
library(ggplot2)
library(pwr)

sturgeon <- read.csv("data/sturgeon.csv")
```

Note that the command to read the data assumes that the data file is in a folder named `data` within the working directory. Adjust as needed.

3.2 Scatter plots

Correlation and regression analysis should always begin with an examination of the data: this is a critical first step in determining whether such analyses are even appropriate for your data. Suppose we are interested in the extent to which length of male sturgeon in the vicinity of The Pas and Cumberland House covaries with weight. To address this question, we look at the correlation between `fklngth` and `rdwght`. Recall that one of the assumptions in correlation analysis is that the relationship between the two variables is linear. To evaluate this assumption, a good first step is to produce a scatterplot.

- Load the data from `sturgeon.csv` in an obk=jcet named `sturgeon`. Make a scatter plot of `rdwght` vs `fklngth` fit with a locally weighted regression (Loess) smoother, and a linear regression line.

```
sturgeon <- read.csv("data/sturgeon.csv")
str(sturgeon)
```

```
## 'data.frame':    186 obs. of  9 variables:
##  $ fklngth : num  37 50.2 28.9 50.2 45.6 ...
##  $ totlngth: num  40.7 54.1 31.3 53.1 49.5 ...
##  $ drlngth : num  23.6 31.5 17.3 32.3 32.1 ...
##  $ rdwght  : num  15.95 NA 6.49 NA 29.92 ...
##  $ age     : int  11 24 7 23 20 23 20 7 23 19 ...
##  $ girth   : num  40.5 53.5 31 52.5 50 54.2 48 28.5 44 39 ...
##  $ sex     : chr  "MALE" "FEMALE" "MALE" "FEMALE" ...
##  $ location: chr  "THE_PAS" "THE_PAS" "THE_PAS" "THE_PAS" ...
##  $ year    : int  1978 1978 1978 1978 1978 1978 1978 1978 1978 1978 ...
```

```
mygraph <- ggplot(
  data = sturgeon[!is.na(sturgeon$rdwght), ], # source of data
  aes(x = fklngth, y = rdwght)
```

```
)
# plot data points, regression, loess trace
mygraph <- mygraph +
  stat_smooth(method = lm, se = FALSE, color = "green") + # add linear regression, but no
  stat_smooth(color = "red", se = FALSE) + # add loess
  geom_point() # add data points

mygraph # display graph
```

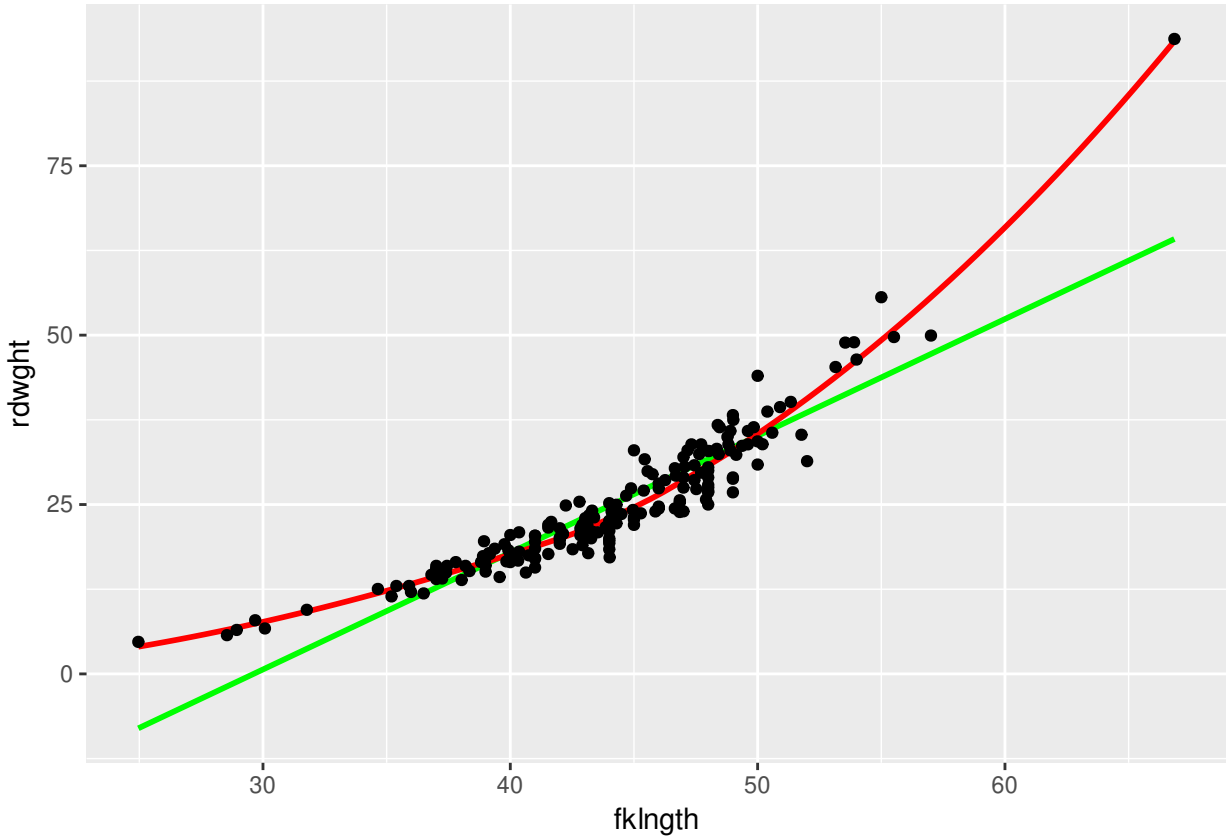


Figure 3.1: Scatter plot of Weight as a function of length in sturgeons

- Does this curve suggest a good correlation between the two? Based on visual inspection, does the relationship between these two variables appear linear?

There is some evidence of nonlinearity, as the curve appears to have a positive second derivative (concave up). This notwithstanding, it does appear the two variables are highly correlated.

- Redo the scatterplot, but after logtransformation of both axes.

```
# apply log transformation on defined graph
mygraph + scale_x_log10() + scale_y_log10()
```

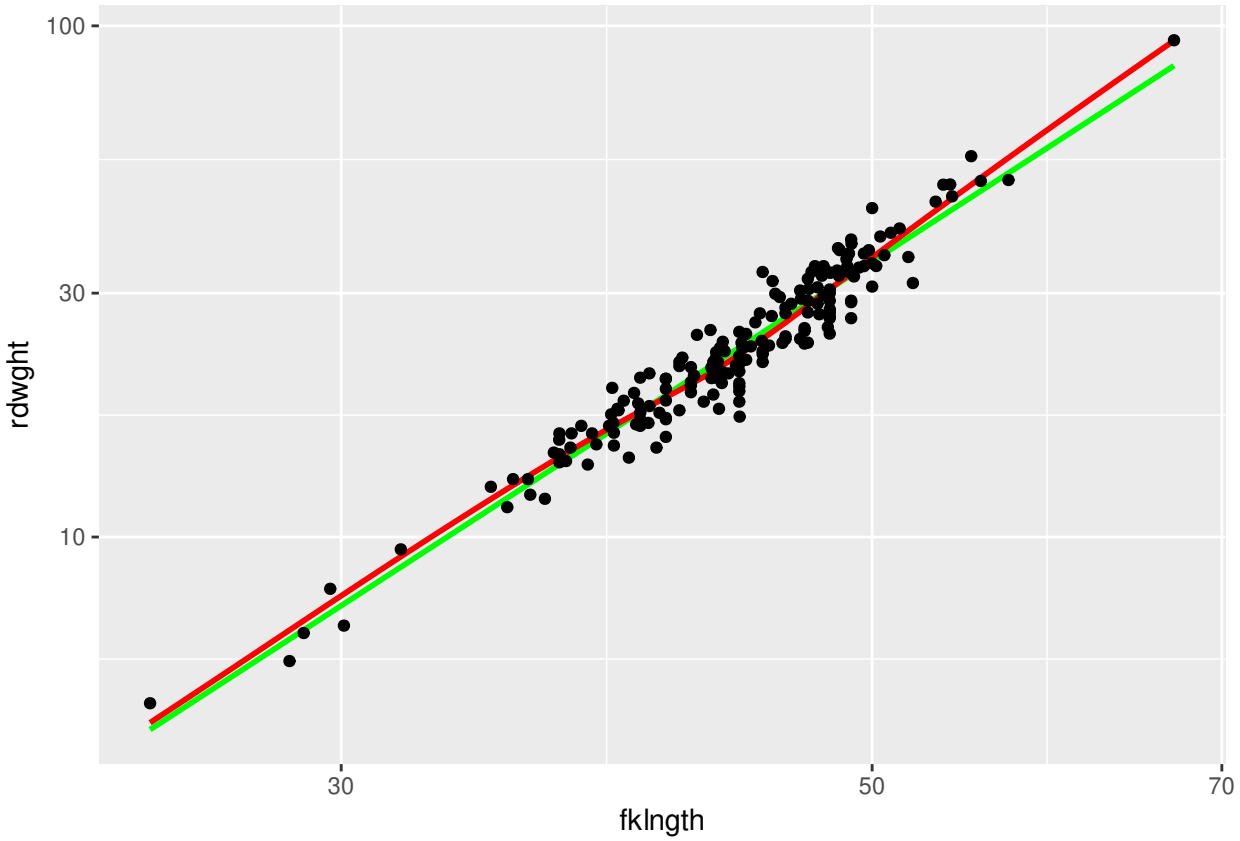


Figure 3.2: Plot weight-length in sturgeon using a log scale

Compare the diagrams before and after the transformation (Figs **??** and **??**). Since the relationship is more linear after transformation, correlation analysis should be done on the transformed data

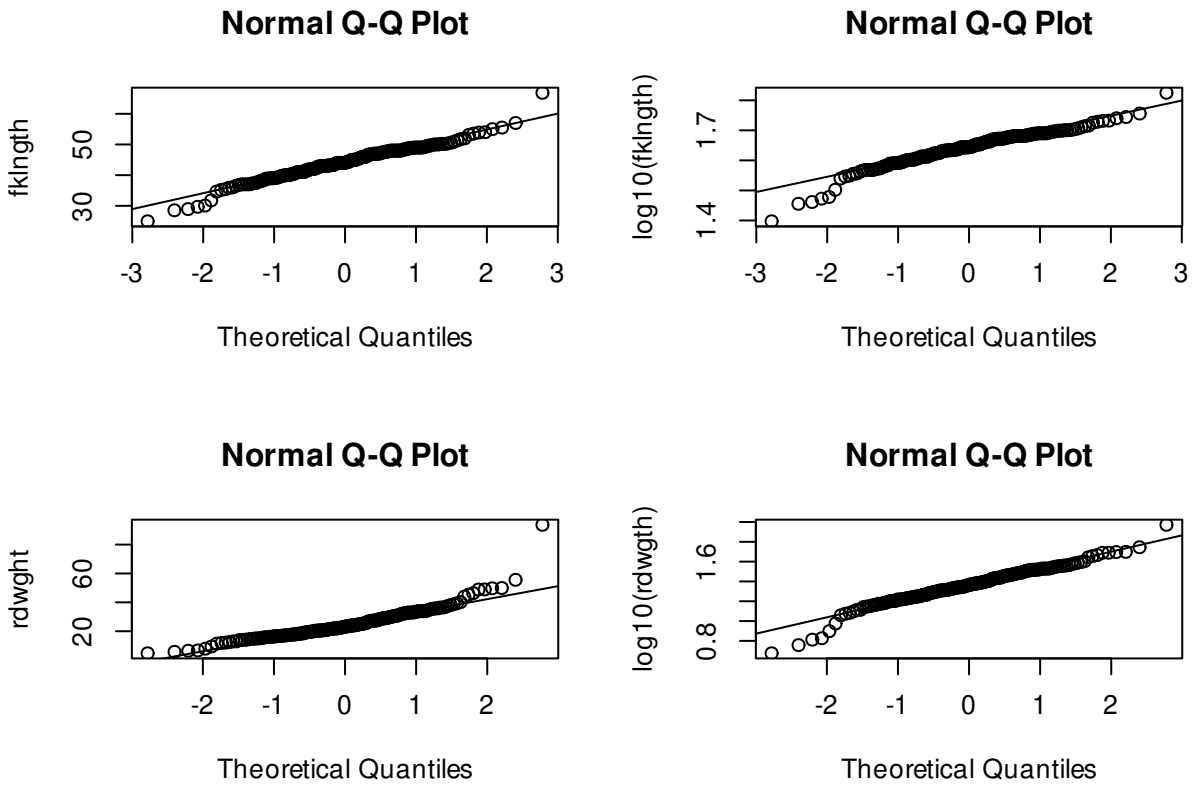
3.3 Data transformations and the product-moment correlation

Recall that another assumption underlying significance testing of the product-moment correlation is that the distribution of the two variables in question is bivariate normal. We can test to see whether each of the two variables are normally distributed using the same procedures outlined in the exercise on two-sample comparisons. If the two variables are each normally distributed, then one is usually (relatively) safe in assuming the joint distribution is normal, although this needn't necessarily be true.

- Examine the distribution of the 4 variables (the two original variables and the log-transformed variables). What do you conclude from visual inspection of these plots?

The following graph contains the 4 QQ plots (`qqplot()`). It was produced by the code below that starts with the `par()` command to ensure that all 4 plots would appear together on the same page in 2 rows and 2 columns:

```
par(mfrow = c(2, 2)) # split graph in 4 (2 rows, 2 cols) filling by rows
qqnorm(sturgeon$fklngth, ylab = "fklngth")
qqline(sturgeon$fklngth)
qqnorm(log10(sturgeon$fklngth), ylab = "log10(fklngth)")
qqline(log10(sturgeon$fklngth))
qqnorm(sturgeon$rdwght, ylab = "rdwght")
qqline(sturgeon$rdwght)
qqnorm(log10(sturgeon$rdwght), ylab = "log10(rdwgth)")
qqline(log10(sturgeon$rdwght))
```




```
par(mfrow = c(1, 1)) # redefine plotting area to 1 plot
```

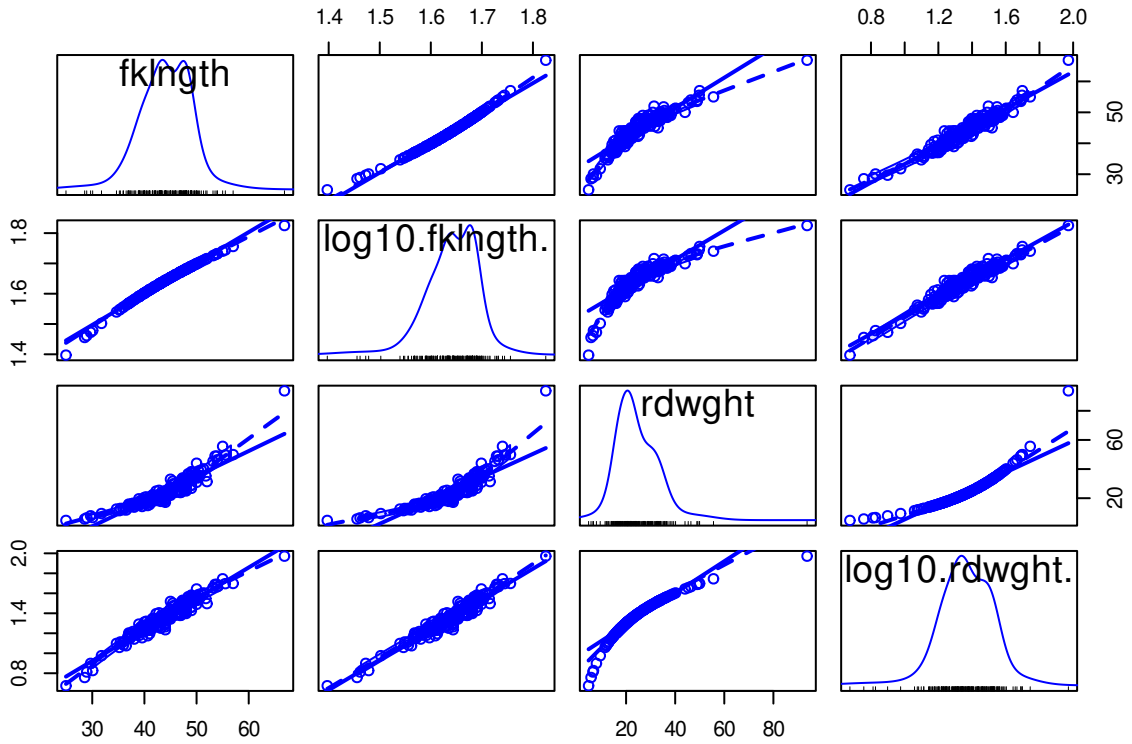
None of these distributions are perfectly normal, but deviations are mostly minor.

- To generate a scatterplot matrix of all pairs of variables, with linear regression and lowess traces, you can use `scatterplotMatrix` from `car` ‘r emo::ji(“package”).

```
scatterplotMatrix(
  ~ fklngth + log10(fklngth) + rdwght + log10(rdwght),
  data = sturgeon,
  smooth = TRUE, diagonal = "density"
```

```
)
```

```
## Warning in applyDefaults(diagonal, defaults = list(method =
## "adaptiveDensity"), : unnamed diag arguments, will be ignored
```



- Next, calculate the Pearson product-moment correlation between each pair (untransformed and log transformed) using the `cor()` command. However, to do this, it will be easier if you first add your transformed data as columns in the sturgeon data frame.

```
sturgeon$lfklngth <- with(sturgeon, log10(fklngth))
sturgeon$lrdwght <- log10(sturgeon$rdwght)
```

Then you can get the correlation matrix by:

```
cor(sturgeon[, c("fklngth", "lfklngth", "lrdwght", "rdwght")], use = "complete.obs")
```

Note the `use="complete.obs"` parameter. It tells R to keep only lines of the data frame where all variables were measured. If there are missing data, some lines will be removed, but correlations will be calculated for the same subset of cases for all pairs of variables. One could use, instead, `use="pairwise.complete.obs"` , to tell R to only eliminate observations when values are missing for this particular pair of variables. In this situation, if there are missing values in the data frame, the sample size for pairwise correlations will vary. In general, I recommend you use the option

`use="complete.obs"`, unless you have so many missing values that it eliminates the majority of your data.

- Why is the correlation between the untransformed variables smaller than between the transformed variables?

```
cor(sturgeon[, c("fklngth", "lfklngth", "lrdwght", "rdwght")], use = "complete.obs")
```

```
##            fklngth  lfklngth   lrdwght    rdwght
## fklngth  1.0000000 0.9921435 0.9645108 0.9175435
## lfklngth 0.9921435 1.0000000 0.9670139 0.8756203
## lrdwght  0.9645108 0.9670139 1.0000000 0.9265513
## rdwght   0.9175435 0.8756203 0.9265513 1.0000000
```

Several things should be noted here.

1. the correlation between fork length and round weight is high, regardless of which variables are used: so as might be expected, heavier fish are also longer, and vice versa
2. the correlation is greater for the transformed variables than the untransformed variables.

Why? Because the correlation coefficient is inversely proportional to the amount of scatter around a straight line. If the relationship is curvilinear (as it is for the untransformed data), the scatter around a straight line will be greater than if the relationship is linear. Hence, the correlation coefficient will be smaller.

3.4 Testing the significance of correlations and Bonferroni probabilities

It's possible to test the significance of individual correlations using the commands window. As an example, let's try testing the significance of the correlation between lfklngth and rdwght (the smallest correlation in the above table).

- In the R script window, enter the following to test the correlation between `lfkgnth` and `rdwght` :

```
cor.test(
  sturgeon$lfklngth, sturgeon$rdwght,
  alternative = "two.sided",
  method = "pearson"
)
```

```
##
##  Pearson's product-moment correlation
##
## data:  sturgeon$lfklngth and sturgeon$rdwght
## t = 24.322, df = 180, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
```

```
## 95 percent confidence interval:
##  0.8367345 0.9057199
## sample estimates:
##       cor
## 0.8756203
```

We see here that the correlation is highly significant ($p < 2.2e-16$), which is no surprise given how high the correlation coefficient is (0.8756).

It's important to bear in mind that when you are estimating correlations, the probability of finding any one correlation that is *"significant"* by pure chance increases with the number of pairwise correlations examined. Suppose, for example, that you have five variables; there are then a total of 10 possible pairwise correlations, and from this set, you would probably not be surprised to find at least one that is "significant" purely by chance. One way of avoiding the problem is to adjust individual α levels for pairwise correlations by dividing by the number of comparisons, k, such that: $\alpha' = \frac{\alpha}{k}$ (Bonferroni probabilities), i.e. if initially, $\alpha = 0.05$ and there are a total of 10 comparisons, then $\alpha' = 0.005$.

In the above example where we examined correlations between `fklngth` and `rdwght` and their `log`, it would be appropriate to adjust the α at which significance is tested by the total number of correlations in the matrix (in this case, 6, so $\alpha' = 0.0083$). Does your decision about the significance of the correlation between `lfklngth` and `rdwght` change?

3.5 Non-parametric correlations: Spearman's rank and Kendall's τ

The analysis done with the sturgeon data in the section above suggests that one of the assumptions of correlation, namely, bivariate normality, may not be valid for `fklngth` and `rdwght` nor for the log transforms of these variables. Finding an appropriate transformation is sometimes like looking for a needle in a haystack; indeed, it can be much worse simply because for some distributions, there is no transformation that will normalize the data. In such cases, the best option may be to go to a non-parametric analysis that does not assume bivariate normality or linearity. All such correlations are based on the ranks rather than the data themselves: two options available in R are Spearman and Kendall's τ (tau).

- Test the correlation between `fklngth` and `rdwght` using both the Spearman and Kendall's tau. The following commands will produce the correlations:

```
cor.test(
  sturgeon$lfklngth, sturgeon$rdwght,
  alternative = "two.sided",
  method = "spearman"
)
```

```
## Warning in cor.test.default(sturgeon$lfklngth, sturgeon$rdwght, alternative =
## "two.sided", : Cannot compute exact p-value with ties
```

```
##
```

```
##  Spearman's rank correlation rho
##
## data:  sturgeon$lfklngth and sturgeon$rdwght
## S = 47971, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.9522546
```

```
cor.test(
  sturgeon$lfklngth, sturgeon$rdwght,
  alternative = "two.sided",
  method = "kendall"
)
```

```
##
##  Kendall's rank correlation tau
##
## data:  sturgeon$lfklngth and sturgeon$rdwght
## z = 16.358, p-value < 2.2e-16
## alternative hypothesis: true tau is not equal to 0
## sample estimates:
##       tau
## 0.8208065
```

Contrast these results with those obtained using the Pearson product-moment correlation. Why the difference?

Test the non-parametric correlations on pairs of the transformed variables. You should immediately note that the non-parametric correlations are identical for untransformed and transformed variables. This is because we are using the ranks, rather than the raw data, and the rank ordering of the data does not change when a transformation is applied to the raw values.

Note that the correlations for Kendall's tau (0.820) are lower than for the Spearman rank (0.952) correlation. This is because Kendall's gives more weight to ranks that are far apart, whereas Spearman's weights each rank equally. Generally, Kendalls's is more appropriate when there is more uncertainty about the reliability of close ranks.

The sturgeons in this sample were collected using nets and baited hooks of a certain size. What impact do you think this method of collection had on the shapes of the distributions of `fklngth` and `rdwght` ? Under these circumstances, do you think correlation analysis is appropriate at all?

Note that correlation analysis assumes that each variable is *randomly sampled*. In the case of sturgeon, this is not the case: baited hooks and nets will only catch sturgeon above a certain minimum size. Note that in the sample, there are no small sturgeons, since the fishing gear targets only larger fish. Thus, we should be very wary of the correlation coefficients associated with our analysis, as the inclusion of smaller fish may well change our estimate of these correlations.

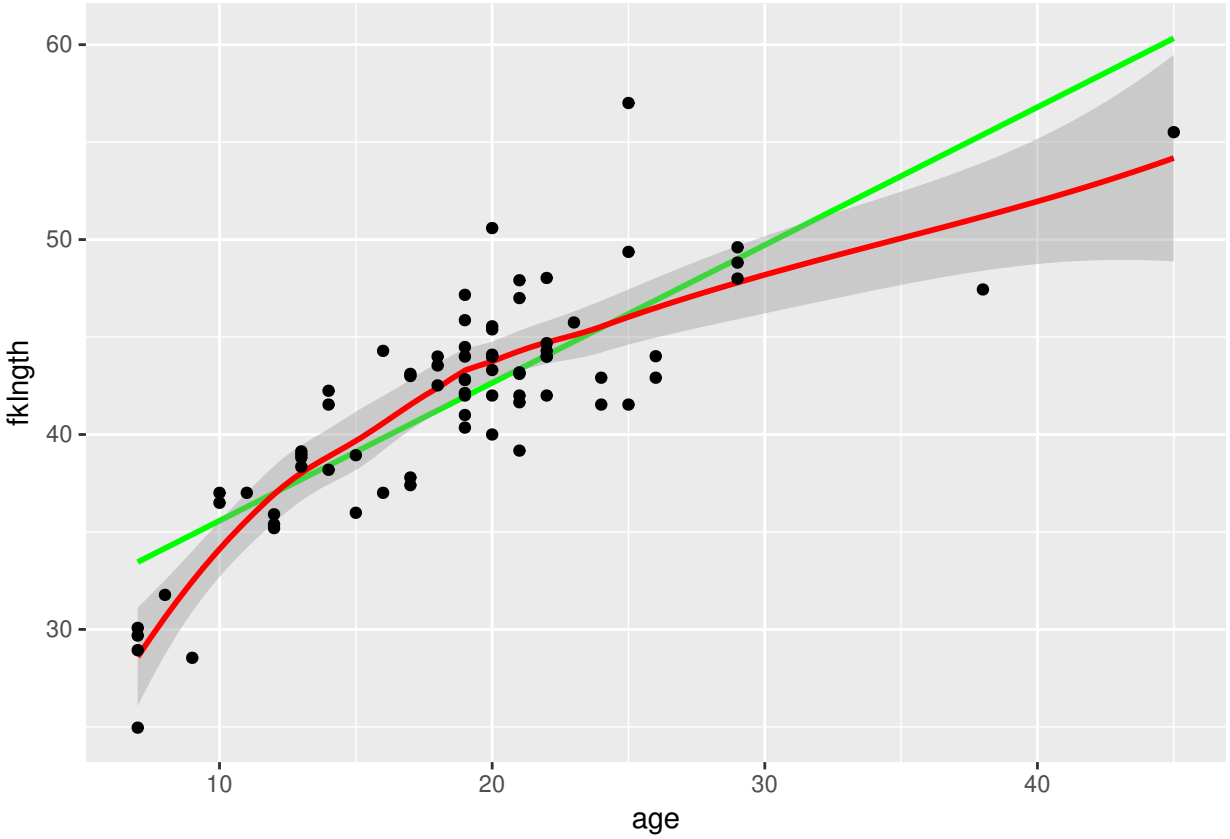
3.6 Simple linear regression

In correlation analysis we are interested in how pairs of variables covary: However, in regression analysis, we are attempting to estimate a model that predicts a variable (the dependent variable) from another variable (the independent variable).

As with any statistical analysis, the best way to begin is by looking at your data. If you are interested in the relationship between two variables, say, Y and X, produce a plot of Y versus X just to get a "feel" for the relationship.

- The data file `sturgeon.csv` contains data for sturgeons collected from 1978-1980 at Cumberland House, Saskatchewan and The Pas, Manitoba. Make a scatterplot of `fklngth` (the dependent variable) versus `age` (the independent variable) for males and add a linear regression and a loess smoother. What do you conclude from this plot?

```
sturgeon.male <- subset(sturgeon, subset = sex == "MALE")
mygraph <- ggplot(
  data = sturgeon.male, # source of data
  aes(x = age, y = fklngth)
) # aesthetics: y=fklngth, x=rdwght
# plot data points, regression, loess trace
mygraph <- mygraph +
  stat_smooth(method = lm, se = FALSE, color = "green") + # add linear regression, but no
  stat_smooth(color = "red") + # add loess
  geom_point() # add data points
mygraph # display graph
```



This suggests that the relationship between age and fork length is not linear.

Suppose that we want to know the growth rate of male sturgeon. One estimate (perhaps not a very good one) of the growth rate is given by the slope of the fork length - age regression.

First, let's run the regression with the lm() command, and save its results in an object called `RegModel.1`.

```
RegModel.1 <- lm(fklngth ~ age, data = sturgeon.male)
```

Nothing appears on the screen, but don't worry, it all got saved in memory. To see the statistical results, type:

```
summary(RegModel.1)
```

```
##
## Call:
## lm(formula = fklngth ~ age, data = sturgeon.male)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.4936 -2.2263  0.1849  1.7526 10.8234
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 28.50359    1.16873   24.39   <2e-16 ***
## age          0.70724    0.05888   12.01   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.307 on 73 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.664,  Adjusted R-squared:  0.6594
## F-statistic: 144.3 on 1 and 73 DF,  p-value: < 2.2e-16
```

R output gives you:

1. `Call:` A friendly reminder of the model fitted and the data used.
2. `Residuals:` General statistics about the residuals around the fitted model.
3. `Coefficients:` Fitted model parameter estimates, standard errors, t values and associated probabilities.
4. `Residual standard error:` Square root of the residual variance.
5. `Multiple R-squared:` Coefficient of determination. It corresponds to the proportion of the total variance of the dependent variable that is accounted for by the regression (i.e. by the independent variable)
6. `Adjusted R-squared:` The adjusted R-squared accounts for the number of parameters in the model. If you want to compare the performance of several models with different numbers of parameters, this is the one to use
7. `F-statistic:` This is the test of the overall significance of the model. In the simple regression case, this is the same as the test of the slope of the regression.

The estimated regression equation is therefore:

$$Fklngth = 28.50359 + 0.70724 * age$$

Given the highly significant F-value of the model (or equivalently the highly significant t-value for the slope of the line), we reject the null hypothesis that there is no relationship between fork length and age.

3.6.1 Testing regression assumptions

Simple model I regression makes four assumptions:

1. the X variable is measured without error;
2. the relationship between Y and X is linear;
3. that for any value of X, the Y's are independently and normally distributed;
4. the variance of Y for fixed X is independent of X.

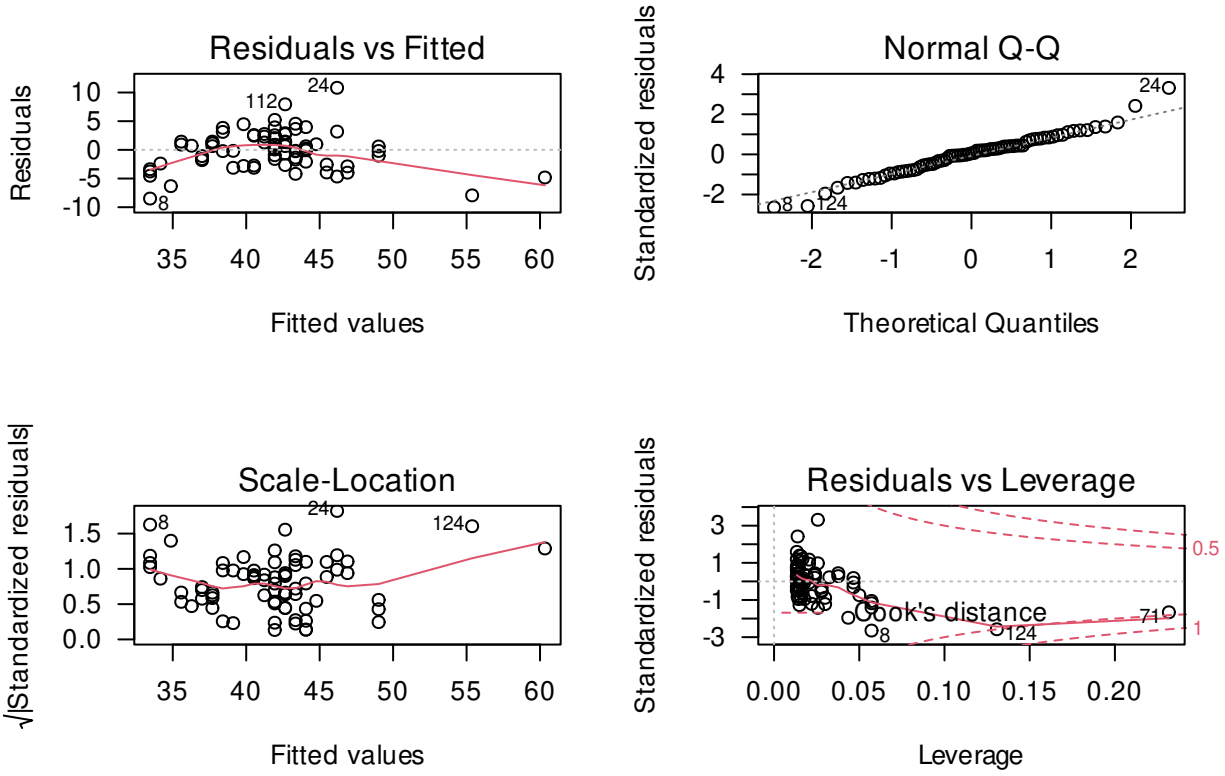
Having done the regression, we can now test the assumptions. For most biological data, the first assumption is almost never valid; usually there is error in both Y and X. This means that in general, slope estimates are biased, but predicted values are unbiased. However, so long as the error in X is small relative to the range of X in your data, the fact that X has an associated error is not likely to influence the outcome dramatically. On the other hand, if there is substantial error

in X, regression results based on a model I regression may give poor estimates of the functional relationship between Y and X. In this case, more sophisticated regression procedures must be employed which are, unfortunately, beyond the scope of this course.

The other assumptions of a model I regression can, however, be tested, or at least evaluated visually. The `plot()` command can display diagnostics for linear models.

```
par(mfrow = c(2, 2), las = 1)
plot(RegModel.1)
```

The `par()` command is used here to tell R to display 2 rows and 2 columns of graphs per page (there are 4 diagnostic graphs for linear models generated automatically), and the last statement is to tell R to rotate the labels of the Y axis so that they are perpendicular to the Y axis. (Yes, I know, this is not at all obvious.)



You will get:

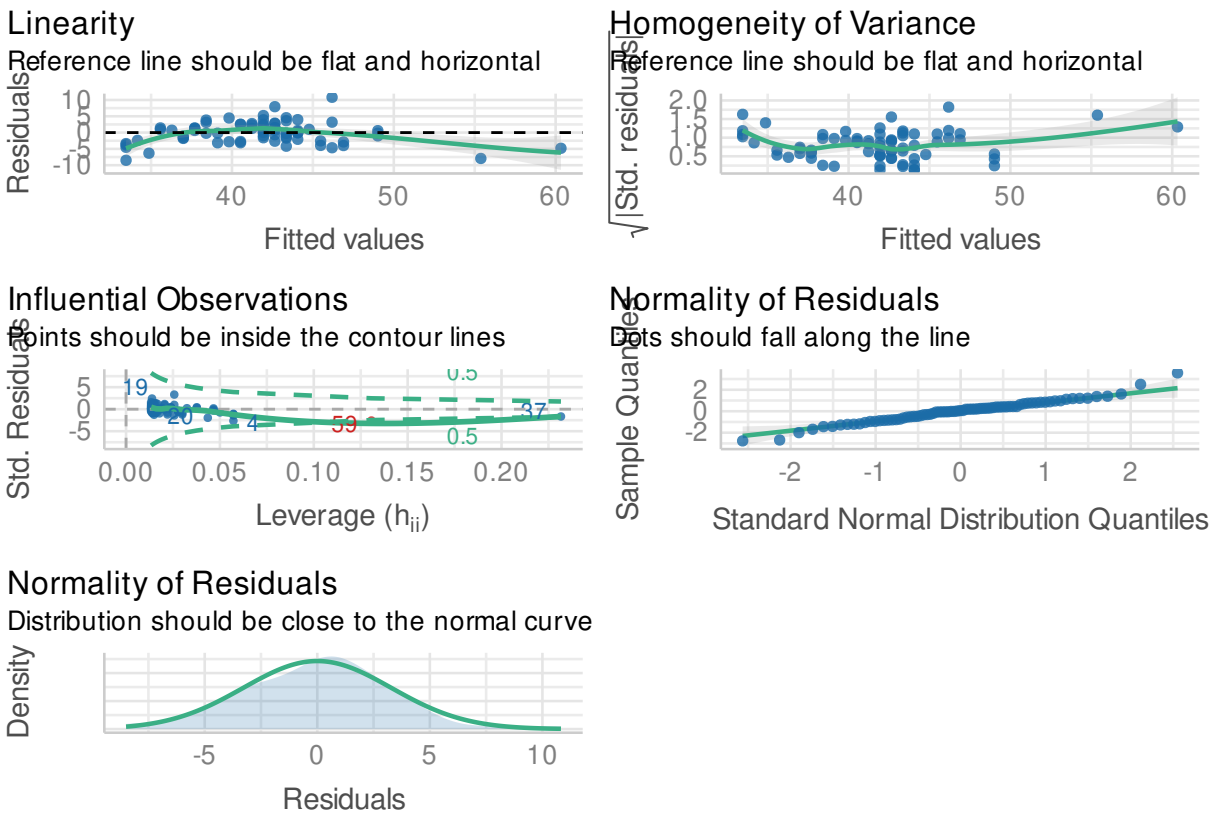
1. **Upper left** tell you about linearity, normality, and homoscedasticity of the residuals. It shows the deviations around the regression vs the predicted values. Remember that the scatterplot (fklngth vs age) suggested that the relationship between fork length and age is not linear. Very young and very old sturgeons tended to fall under the line, and fish of average age tended to be a bit above the line. This is exactly what the residual vs fitted plot shows. The red line is a lowess trace through these data. If the relationship was linear, it would be approximately flat and close to 0. The scatter of residuals tells you a bit about their normality and homoscedasticity, although this graph is not the best way to look at these properties. The next two are better.

2. **Upper right** is to assess the normality of the residuals. It is a QQ plot of the residuals . If the residuals were normally distributed, they would fall very close to the diagonal line. Here, we see it is mostly the case, except in the tails
3. **Bottom left** titled Scale-Location, helps with assessing homoscedasticity. It plots the square root of the absolute value of the standardized residual (residual divided by the standard error of the residuals, this scales the residuals so that their variance is 1) as a function of the fitted value. This graph can help you visualize whether the spread of the residuals is constant or not. If residuals are homoscedastic, then the average will not change with increasing fitted values. Here, there is slight variability, but it is not monotonous (i.e. it does not increase or decrease systematically) and there is no strong evidence against the assumption of homoscedasticity.
4. **Bottom right** plots the residuals as a function of leverage and can help detecting the presence of outliers or points that have a very strong influence on the regression results. The leverage of a point measures how far it is from the other points, but only with respect to the independent variable. In the case of simple linear regression, it is a function of the difference between the observation and the mean of the independent variable. You should look more closely at any observation with a leverage value that is greater than: $2(k+1)/n$, where \$$k$ is the number of independent variables (here 1), and n is the number of observations. In this case there is 1 independent variable, 75 observations, and points with a leverage higher than 0.053 may warrant particular scrutiny. The plot also gives you information about how the removal of a point from the data set would change the predictions. This is measured by the Cook's distance, illustrated by the red lines on the plot. A data point with a Cook distance larger than 1 has a large influence.

Note that R automatically labels the most extreme cases on each of these 4 plots. It does not mean that these cases are outliers, or that you necessarily need be concerned with them. In any data set, there will always be a minimum and a maximum residual.

The R package `performance` offers a new and updated version of those graphs with colours and more plots to help visually assess the assumptions with the function `model_check()`

```
check_model(RegModel.1)
```



So, what is the verdict about the linear regression between `fklngth` and `age` ? It fails the linearity, possibly fails the normality, passes homoscedasticity, and this does not seem to be too strongly affected by some bizarre points.

3.6.2 Formal tests of regression assumptions

In my practice, I seldom use formal tests of regression assumptions and mostly rely on graphs of the residuals to guide my decisions. To my knowledge, this is what most biologists and data analysts do. However, in my early analyst life I was not always confident that I was interpreting these graphs correctly and wished that I had a formal test or a statistic quantifying the degree of deviation from the regression assumptions.

The `lmtest` R package, not part of the base R installation, but available from CRAN, contains a number of tests for linearity and homoscedasticity. And one can test for normality using the Shapiro-Wilk test seen previously.

First, you need to load (and maybe install) the `lmtest` package.

```
library(lmtest)
```

Run the following commands

```
bptest(RegModel.1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  RegModel.1
## BP = 1.1765, df = 1, p-value = 0.2781
```

The Breusch-Pagan test examines whether the variability of the residuals is constant with respect to increasing fitted values. A low p value is indicative of heteroscedasticity. Here, the p value is high, and supports my visual assessment that the homoscedasticity assumption is met by these data.

```
dwtest(RegModel.1)
```

```
##
##  Durbin-Watson test
##
## data:  RegModel.1
## DW = 2.242, p-value = 0.8489
## alternative hypothesis: true autocorrelation is greater than 0
```

The Durbin-Watson test can detect serial autocorrelation in the residuals. Under the assumption of no autocorrelation, the D statistic is 2. This test can detect violation of independence of observations (residuals), although it is not foolproof. Here there is no problem identified.

```
resettest(RegModel.1)
```

```
##
##  RESET test
##
## data:  RegModel.1
## RESET = 14.544, df1 = 2, df2 = 71, p-value = 5.082e-06
```

The RESET test is a test of the assumption of linearity. If the linearity assumption is met, the RESET statistic will be close to 1. Here, the statistic is much larger (14.54), and very highly significant. This confirms our visual assessment that the relationship is not linear.

```
shapiro.test(residuals(RegModel.1))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(RegModel.1)
## W = 0.98037, p-value = 0.2961
```

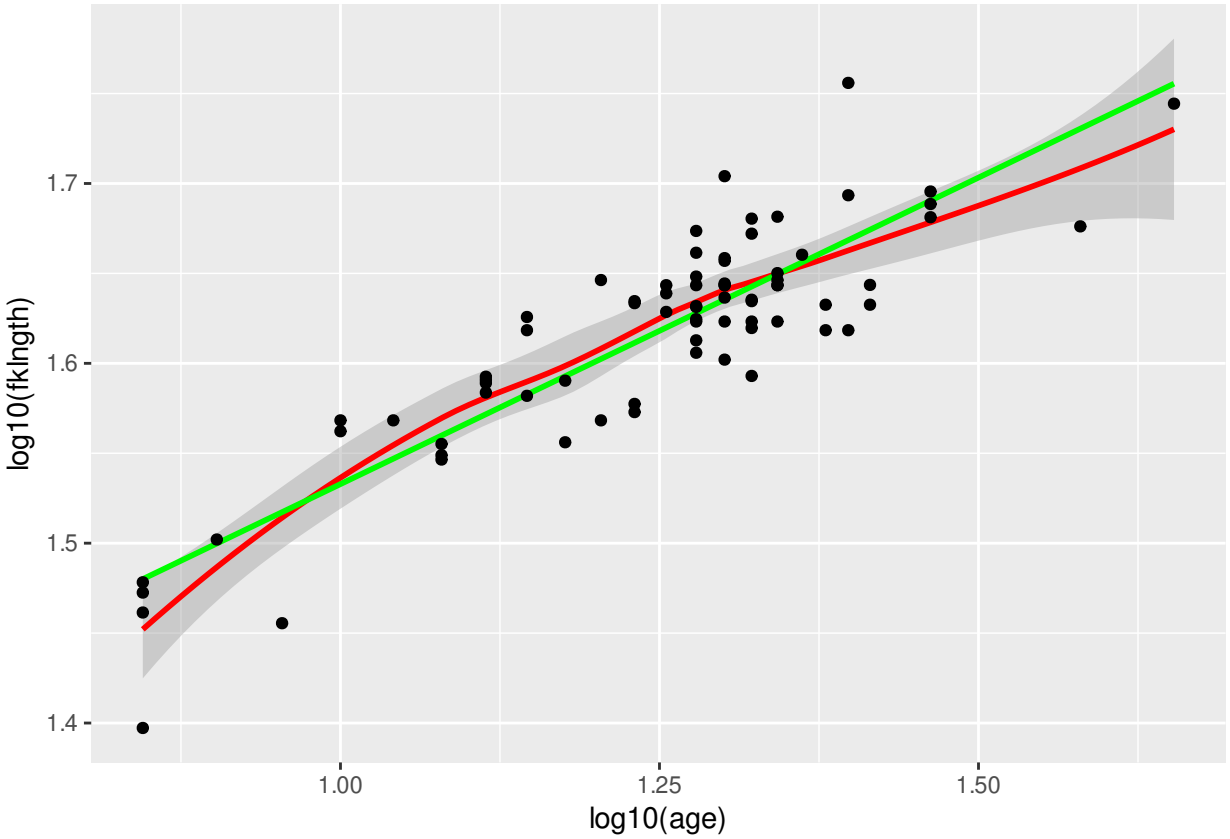
The Shapiro-Wilk normality test on the residual confirms that the deviation from normality of the residuals is not large.

3.7 Data transformations in regression

The analysis above revealed that the linearity assumption underlying regression analysis is not met by the `fklngth` - `age` data. If we want to use regression analysis, data transformations are required:

Let's plot the log-transformed data

```
par(mfrow = c(1, 1), las = 1)
ggplot(
  data = sturgeon.male,
  aes(x = log10(age), y = log10(fklngth))
) +
  geom_smooth(color = "red") +
  geom_smooth(method = "lm", se = FALSE, color = "green") +
  geom_point()
```



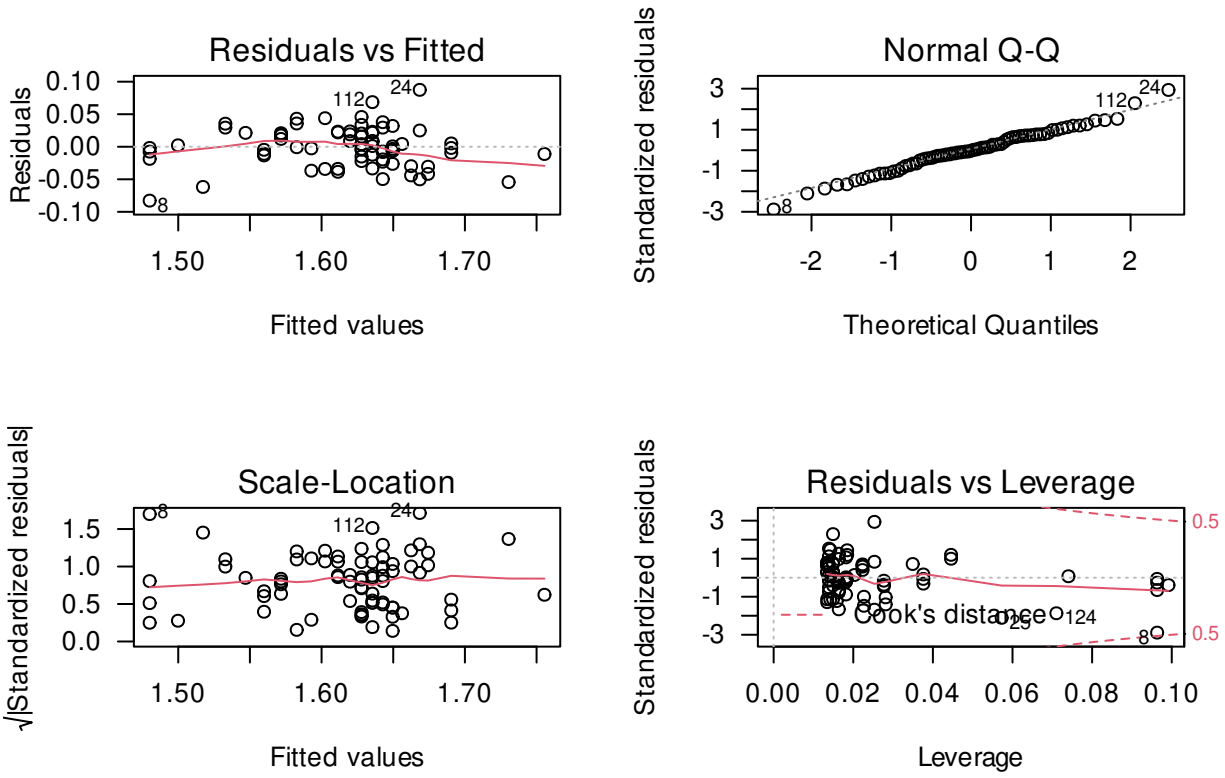
We can fit the linear regression model on the log-transformed variables.

```
RegModel.2 <- lm(log10(fklngth) ~ log10(age), data = sturgeon.male)
summary(RegModel.2)
```

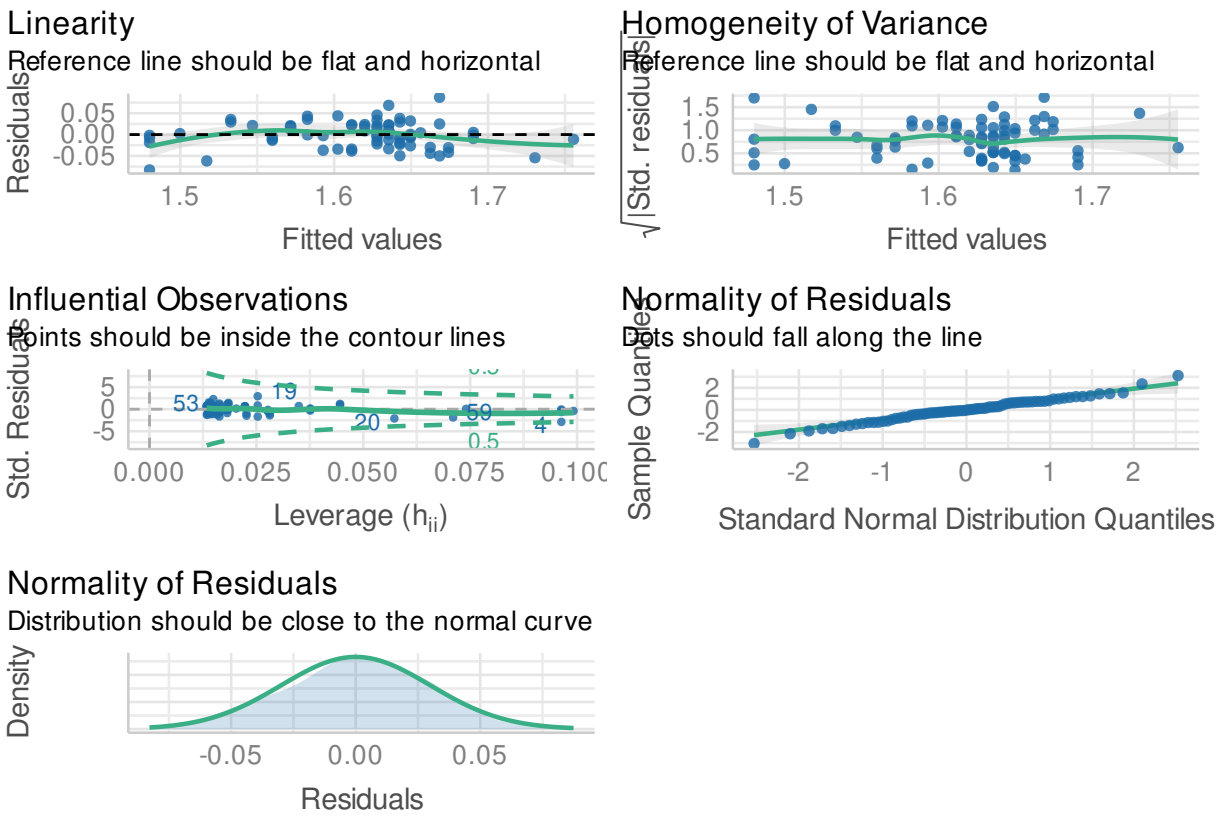
```
##
## Call:
## lm(formula = log10(fklngth) ~ log10(age), data = sturgeon.male)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.082794 -0.016837 -0.000719  0.021102  0.087446
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.19199    0.02723   43.77   <2e-16 ***
## log10(age)   0.34086    0.02168   15.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03015 on 73 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.772,  Adjusted R-squared:  0.7688
## F-statistic: 247.1 on 1 and 73 DF,  p-value: < 2.2e-16
```

Note that by using the log transformed data, the proportion of variation explained by the regression has increased by 10% (from 0.664 to 0.772), a substantial increase. So the relationship has become more linear. Good. Let's look at the residual diagnostic plots:

```
par(mfrow = c(2, 2), las = 1)
plot(RegModel.2)
```




```
check_model(RegModel.2)
```



So things appear a little better than before, although still not ideal. For example, the Residual vs fitted plot still suggests a potential nonlinearity. The QQ plot is nicer than before, indicating that residuals are more normally distributed after the log-log transformation. There is no indication of heteroscedasticity. And, although there are still a few points with somewhat high leverage, none have a Cook's distance above 0.5. It thus seems that transforming data improved things: more linear, more normal, less dependence on extreme data. Do the formal tests support this visual assessment?

```
bptest(RegModel.2)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  RegModel.2
## BP = 0.14282, df = 1, p-value = 0.7055
```

```
dwtest(RegModel.2)
```

```
##
##  Durbin-Watson test
##
## data:  RegModel.2
## DW = 2.1777, p-value = 0.6134
```

```
## alternative hypothesis: true autocorrelation is greater than 0
```

```
resettest(RegModel.2)
```

```
##
##  RESET test
##
## data:  RegModel.2
## RESET = 4.4413, df1 = 2, df2 = 71, p-value = 0.01523
```

```
shapiro.test(residuals(RegModel.2))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(RegModel.2)
## W = 0.98998, p-value = 0.8246
```

Indeed, they do: residuals are still homoscedastic (Breusch-Pagan test), show no autocorrelation (Durbin-Watson test), are normal (Shapiro-Wilk test), and they are more linear (p value of the RESET test is now 0.015, instead of 0.000005). Linearity has improved, but is still violated somewhat.

3.8 Dealing with outliers

In this case, there are no real clear outliers. Yes, observations 8, 24, and 112 are labeled as the most extreme in the last set of residual diagnostic plots. But they are still within what I consider the "reasonable" range. But how does one define a limit to the reasonable? When is an extreme value a real outlier we have to deal with? Opinions vary about the issue, but I favor conservatism.

My rule is that, unless the value is clearly impossible or an error in data entry, I do not delete "outliers"; rather, I analyze all my data. Why? Because, I want my data to reflect natural or real variability. Indeed, variability is often what interests biologists the most.

Keeping extreme values is the fairest way to proceed, but it often creates other issues. These values will often be the main reason why the data fail to meet the assumptions of the statistical analysis. One solution is to run the analysis with and without the outliers, and compare the results. In many cases, the two analyses will be qualitatively similar: the same conclusions will be reached, and the effect size will not be very different. Sometimes, however, this comparison will reveal that the presence of the outliers changes the story. The logical conclusion then is that the results depend on the outliers and that the data at hand are not very conclusive. As an example, let's rerun the analysis after eliminating observations labeled 8, 24, and 112.

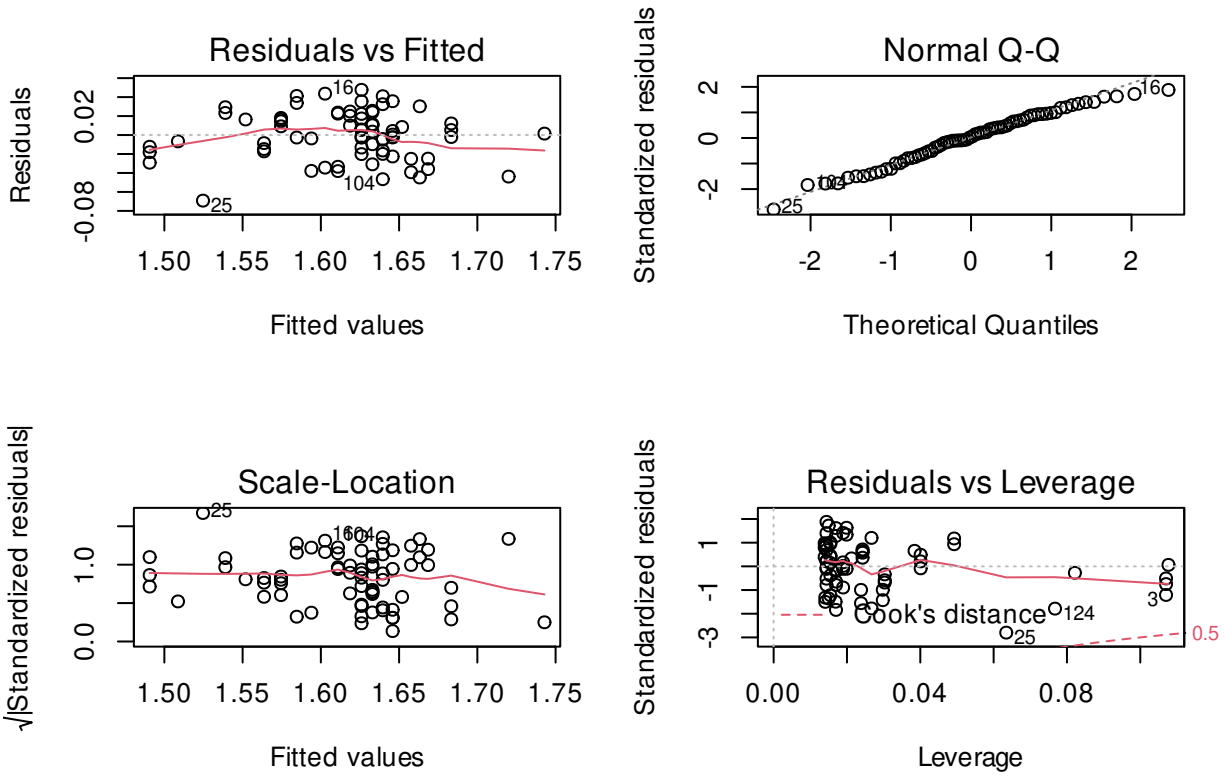
```
RegModel.3 <- lm(log10(fklngth) ~ log10(age), data = sturgeon.male, subset = !(rownames(st
summary(RegModel.3)
```

```
##
## Call:
## lm(formula = log10(fklngth) ~ log10(age), data = sturgeon.male,
##     subset = !(rownames(sturgeon.male) %in% c("8", "24", "112")))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.069163 -0.017390  0.000986  0.018590  0.047647
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.22676    0.02431   50.46   <2e-16 ***
## log10(age)   0.31219    0.01932   16.16   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02554 on 70 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.7885, Adjusted R-squared:  0.7855
## F-statistic:   261 on 1 and 70 DF,  p-value: < 2.2e-16
```

The intercept, slope, and R squared are about the same, and the significance of the slope is still astronomical. Removing the "outliers" has little effect in this case.

As for the diagnostic residual plots and the formal tests of assumptions:

```
par(mfrow = c(2, 2))
plot(RegModel.3)
```




```
bptest(RegModel.3)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  RegModel.3
## BP = 0.3001, df = 1, p-value = 0.5838
```

```
dwtest(RegModel.3)
```

```
##
##  Durbin-Watson test
##
## data:  RegModel.3
## DW = 2.0171, p-value = 0.5074
## alternative hypothesis: true autocorrelation is greater than 0
```

```
resettest(RegModel.3)
```

```
##
##  RESET test
##
```

```
## data:  RegModel.3
## RESET = 3.407, df1 = 2, df2 = 68, p-value = 0.0389
```

```
shapiro.test(residuals(RegModel.3))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(RegModel.3)
## W = 0.98318, p-value = 0.4502
```

No real difference either. Overall, this suggests that the most extreme values do not have undue influence on the results.

3.9 Quantifying effect size in regression and power analysis

Biological interpretation differs from statistical interpretation. Statistically, we conclude that size increase with age (i.e. the slope is positive and different from 0). But this conclusion alone does not tell if the difference between young and old fish is large. The slope and the scatterplot are more informative than the p-value here. The slope (in log-log space) is 0.34. This means that for each unit increase of X (`log10(age)`), there is an increase of 0.34 units of `log10(fklngth)`. In other words, when age is multiplied by 10, fork length is multiplied by about 2 (10^0.34). Humm, length increases more slowly than age. This slope value (0.34) is an estimate of raw effect size. It measure how much length changes with age.

3.9.1 Power to detect a given slope

You can compute power with G*Power for some slope value that you deem of sufficient magnitude to warrant detection.

1. Go to **t Tests: Linear bivariate regression: One group, size of slope**.
2. Select **Post hoc: Compute achieved power- given α, sample size,and effect size**

For example, suppose that sturgeon biologists deem that a slope of 0.1 for the relationship between `log10(fklngth)` and `log10(age)` is meaningful and you wanted to estimate the power to detect such a slope with a sample of 20 sturgeons. Results from the log-log regression contain most of what you need:

```
summary(RegModel.2)
```

```
##
## Call:
## lm(formula = log10(fklngth) ~ log10(age), data = sturgeon.male)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.082794 -0.016837 -0.000719  0.021102  0.087446
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.19199    0.02723   43.77   <2e-16 ***
## log10(age)   0.34086    0.02168   15.72   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03015 on 73 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.772,  Adjusted R-squared:  0.7688
## F-statistic: 247.1 on 1 and 73 DF,  p-value: < 2.2e-16
```

Note the Residual standard error value (0.03015). You will need this. The other thing you need is an estimate of the standard deviation of `log10(age)`. R can (of course) compute it. Be careful, the `sd()` function will return `NA` if there are missing values. You can get around this by adding `na.rm=TRUE` as an argument ot the `sd()` function.

```
sd(log10(sturgeon.male$age), na.rm = TRUE)
```

```
## [1] 0.1616675
```

You can then enter these values (slope to be detected, sample size, alpha, standard deviation of the independent variable) to calculate another quantity that G*Power needs (standard deviation of y) using the Determine panel. Finally you can calculate Power. The filled panels should look like this

Note: The SD of y can't just be taken from the data because if the slope chanages (e.g. H1) then this will change the SD of y. SD y needs to be estimated from the observed scatter around the line and the hypothesized slope).

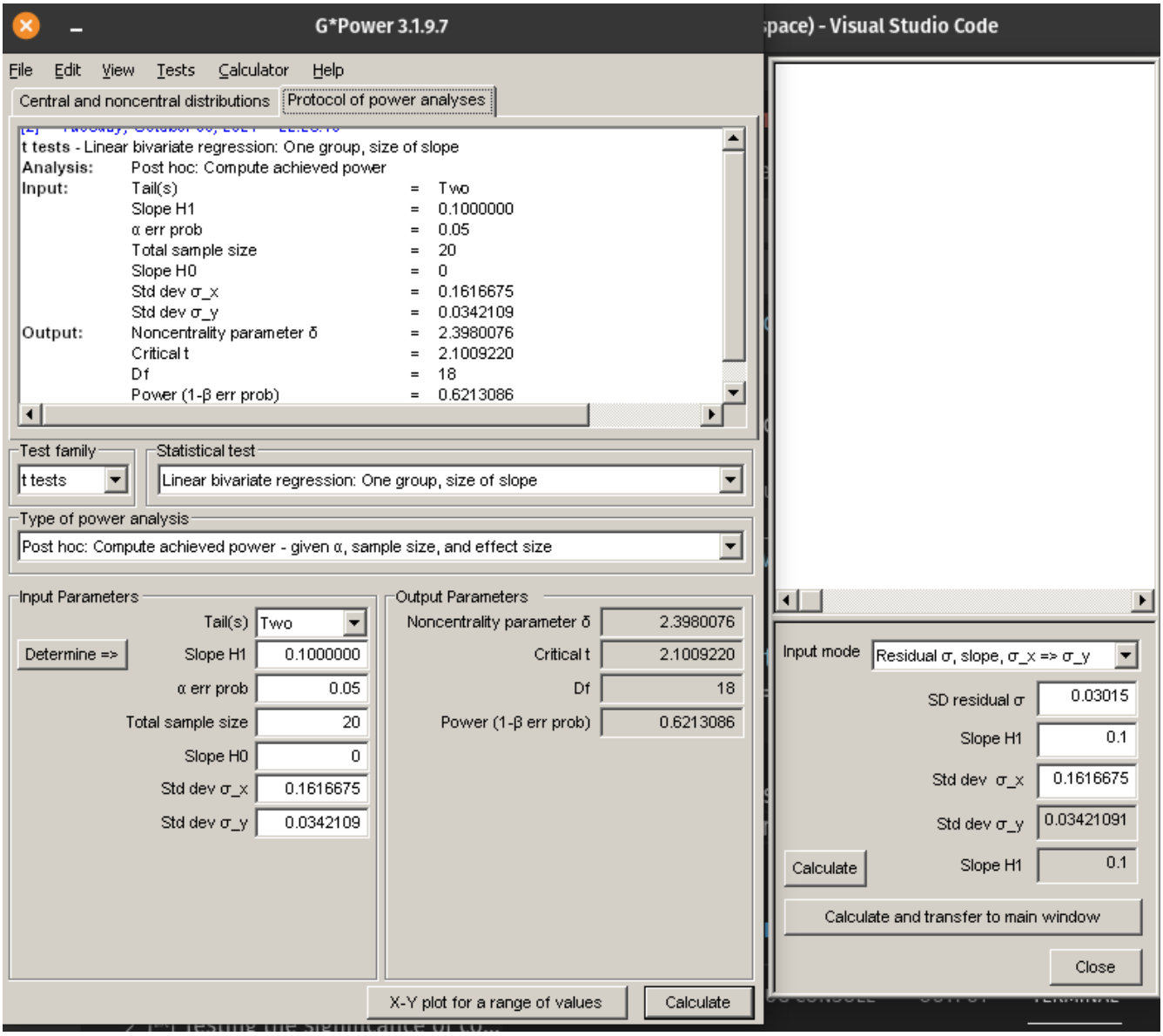


Figure 3.3: Power analysis for age-length in sturgeon with N = 20 and slope = 0.1

Power to detect a significant slope, if the slope is 0.1, variability of data points around the regression is like in our sample, for a sample of 20 sturgeons, with $\alpha = 0.05$ is 0.621. Only about 2/3 of samples of that size would detect a significant effect of `age` on `fklngth`.

In R, you can do the analysis also but we will use another trick to work with the `pwr.t.test()` function. First we, need to estimate the effect size `d`. IN this case `d` is estimated as:

$$d = \frac{b}{s_b\sqrt{n-k-1}}$$

where b is the slope, s_b is the standard error on the slope, n is the number of observations and k is the number of independent variables (1 for simple liner regression).

SE of the slope is 0.02168. The model was fitted using 75 fishes (n=75). We can then estimate `d`.

$$d = \frac{b}{s_b\sqrt{n-k-1}} = \frac{0.1}{0.02168\sqrt{74-1-1}} = 0.54$$

We can simply use the `pwr.t.test()` function to estimate the power.

```
library(pwr)

# analyse de puissance
pwr.t.test(n = 20, d = 0.54, sig.level = 0.05, type = "one.sample")
```

```
##
##      One-sample t test power calculation
##
##               n = 20
##               d = 0.54
##       sig.level = 0.05
##           power = 0.6299804
##     alternative = two.sided
```

You can see that the results is really similar but not exactly the same than with G*power which is normal since we did not use the exact same formula to estimate power.

3.9.2 Sample size required to achieve desired power

To estimate the sample size required to achieve 99% power to detect a slope of 0.1 (in log-log space), with alpha=0.05, you simply change the type of analysis:

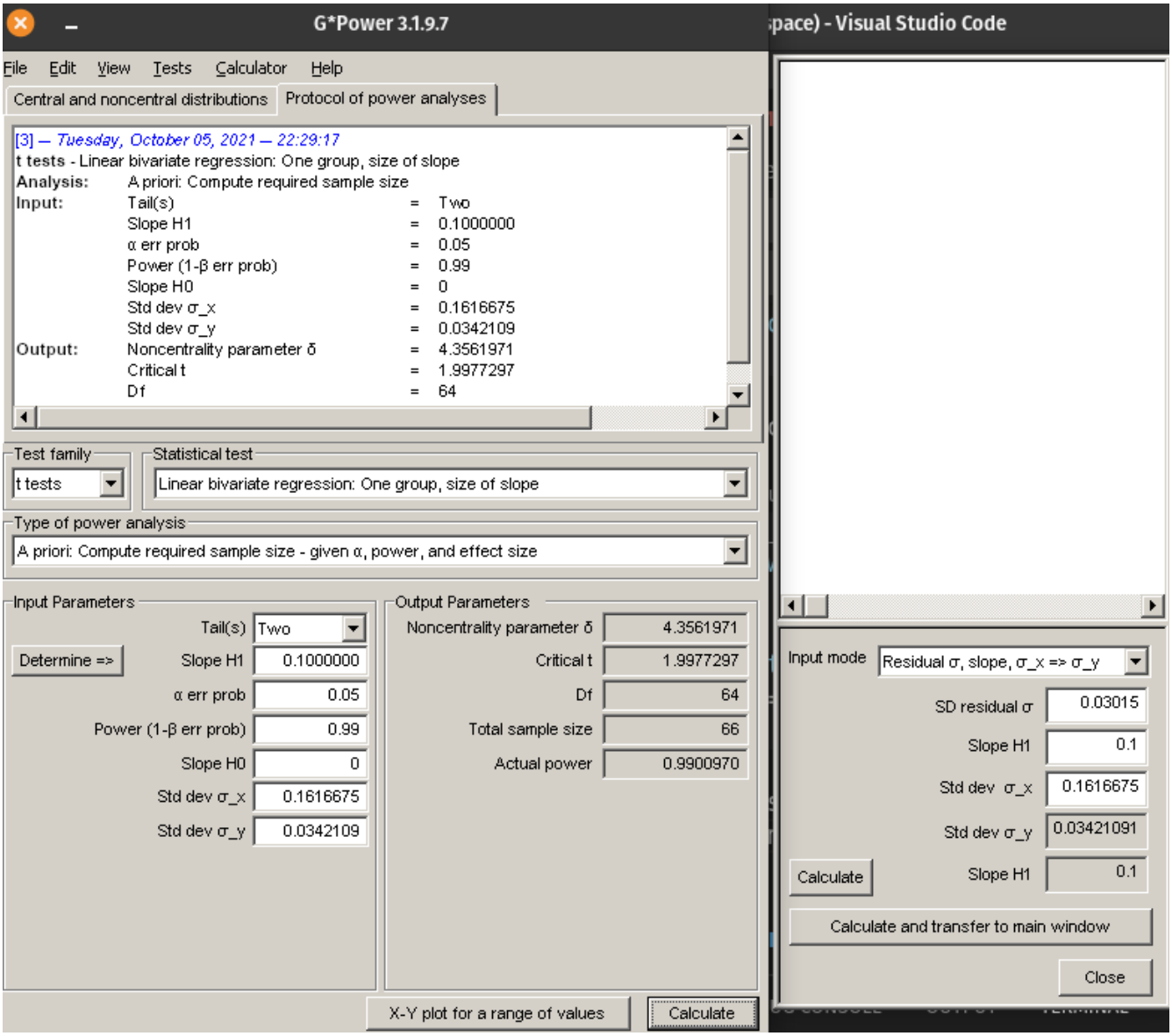


Figure 3.4: A priori power analysis to estimate the sample size needed to have a power of 0.99

In R you can simply do:

```
library(pwr)

# analyse de puissance
pwr.t.test(power = 0.99, d = 0.54, sig.level = 0.05, type = "one.sample")
```

```
##
##      One-sample t test power calculation
##
##               n = 64.96719
```

```
##               d = 0.54
##       sig.level = 0.05
##           power = 0.99
##     alternative = two.sided
```

By increasing sample size to 66, with the same assumptions as before, power increases to 99%.

3.10 Bootstrapping the simple linear regression

A non-parametric test for the intercept and slope of a linear regression can be obtained by bootstrapping.

```
# load boot
library(boot)
# function to obtain regression weights
bs <- function(formula, data, indices) {
  d <- data[indices, ] # allows boot to select sample
  fit <- lm(formula, data = d)
  return(coef(fit))
}
# bootstrapping with 1000 replications
results <- boot(
  data = sturgeon.male,
  statistic = bs,
  R = 1000, formula = log10(fklngth) ~ log10(age)
)
# view results
results
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = sturgeon.male, statistic = bs, R = 1000, formula = log10(fklngth) ~
##     log10(age))
##
##
## Bootstrap Statistics :
##      original       bias    std. error
## t1* 1.1919926  0.002516046  0.03236783
## t2* 0.3408557 -0.001825217  0.02577462
```

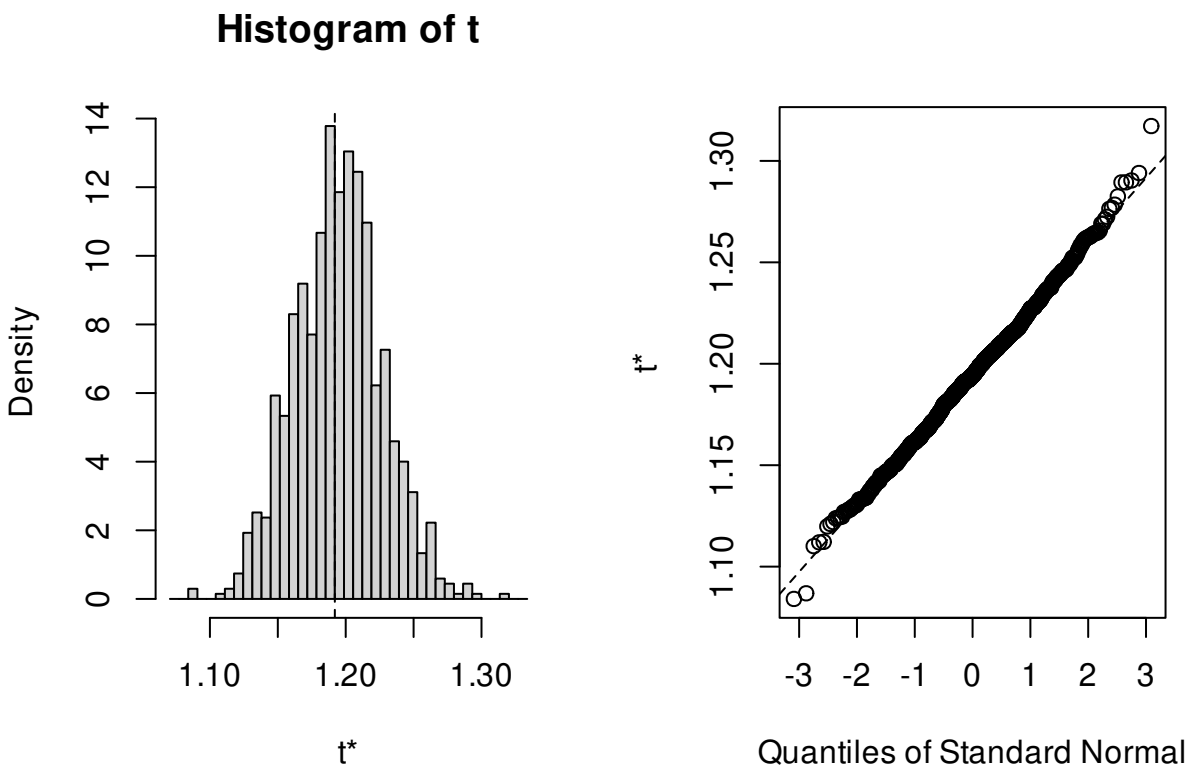
For each parameter in the model (here the intercept is labeled `t1\*` and the slope of the regression line is labeled `t2\*`) , you obtain:

Pour chaque paramètre du modèle (ici l'ordonnée à l'origine est appelée t1* et la pente de la

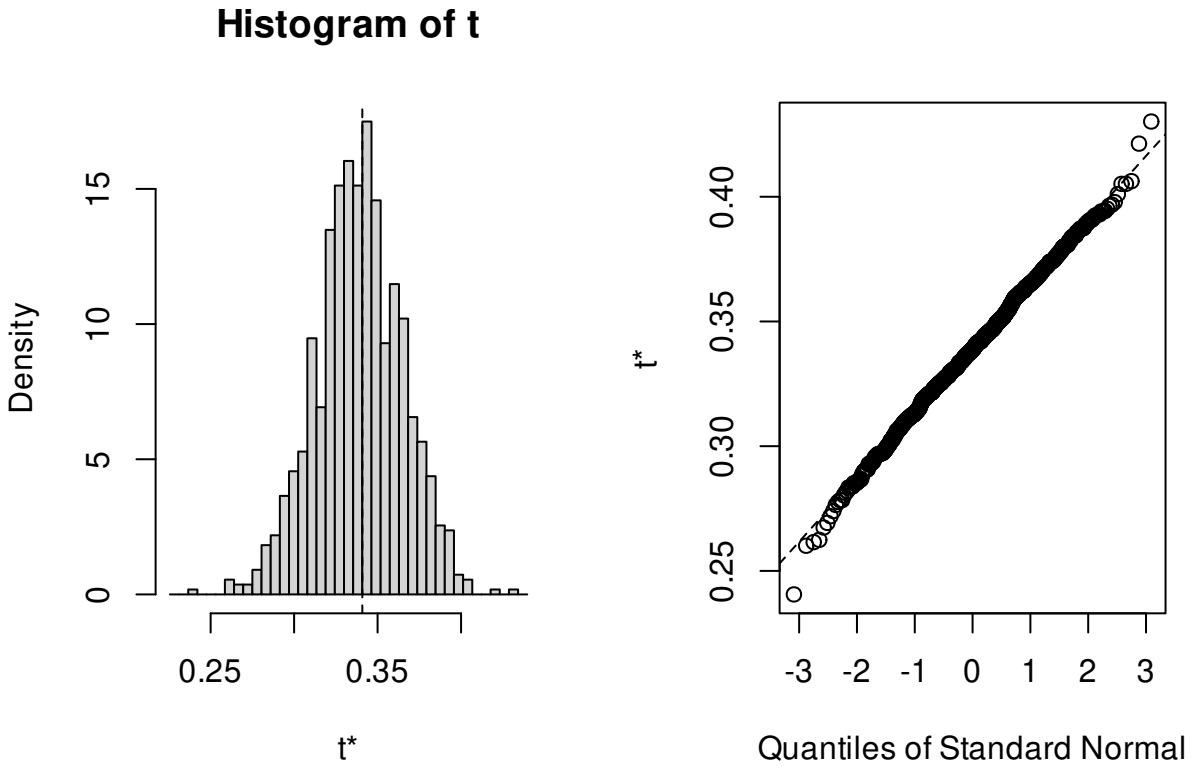
régression `t2\*`), R imprime :

1. `original` original parameter estimate (on all non-bootstrapped data)
2. `bias` the difference between the mean value of all bootstrap estimates and the original value
3. `std. error` standard error of the bootstrap estimate

```
par(mfrow = c(2, 2))
plot(results, index = 1) # intercept
```




```
plot(results, index = 2) # log10(age)
```



The distribution of the bootstrapped estimates is rather Gaussian, with only small deviations in the tails (where it counts for confidence intervals…). One could use the standard error of the bootstrap estimates to calculate a symmetrical confidence interval as mean +- t SE. But, given that R can as easily calculate a bias-corrected adjusted (BCa) confidence interval, or one based on the actual distribution, (Percentile) why not have it do it all:

```
# interval de confiance pour l'ordonnée à l'origine
boot.ci(results, type = "all", index = 1)
```

```
## Warning in boot.ci(results, type = "all", index = 1): bootstrap variances needed
## for studentized intervals
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = results, type = "all", index = 1)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 1.126,  1.253 )   ( 1.122,  1.251 )
##
## Level     Percentile            BCa
## 95%   ( 1.133,  1.262 )   ( 1.123,  1.247 )
```

```
## Calculations and Intervals on Original Scale
## Some BCa intervals may be unstable
```

```
# intervalle de confiance pour la pente
boot.ci(results, type = "all", index = 2)
```

```
## Warning in boot.ci(results, type = "all", index = 2): bootstrap variances needed
## for studentized intervals
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = results, type = "all", index = 2)
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 0.2922,  0.3932 )   ( 0.2926,  0.3953 )
##
## Level     Percentile            BCa
## 95%   ( 0.2864,  0.3891 )   ( 0.2959,  0.3948 )
## Calculations and Intervals on Original Scale
```

Here the 4 types of CI that R managed to calculate are essentially the same. Had data been violating more strongly the standard assumptions (normality, homoscedasticity), then the different methods (Normal, Basic, Percentile, and BCa) would have diverged more. In that case, which one is best? BCa has the favor of most, currently.

Chapitre 4

Two - sample comparisons

After completing this laboratory exercise, you should be able to:

- Use R to visually examine data.
- Use R to compare the means of two normally distributed samples.
- Use R to compare the means of two non-normally distributed samples.
- Use R to compare the means of two paired samples

4.1 R packages and data

For this la b you need:

- R packages:
  - car
  - lmtest
  - boot
  - pwr
  - ggplot2
  - performance
- data:
  - sturgeon.csv
  - skulldat_2020.csv

You need to load the packages in R with `library()` and if need needed install them first with `install.packages()` For the data, load them using the `read.csv()` function.

4.2 Visual examination of sample data

One of the first steps in any type of data analysis is to visualize your data with plots and summary statistics, to get an idea of underlying distributions, possible outliers, and trends in your data. This often begins with plots of the data, such as histograms, probability plots, and box plots, that allow you to get a feel for whether your data are normally distributed, whether they are correlated one to the other, or whether there are any suspicious looking points that may lead you to go back to the original data file to check for errors.

Suppose we want to test the null hypothesis that the size, as indexed by fork length (`fklngth` in file `sturgeon.csv` - the length, in cm, from the tip of the nose to the base of the fork in the caudal fin), of sturgeon at The Pas and Cumberland House is the same. To begin, we have a look at the underlying distributions of the sample data to get a feel for whether the data are normally distributed in each sample. We will not actually test for normality at this point; the assumption of normality in parametric analyses refers always to the residuals and not the raw data themselves. However, if the raw data are non-normally distributed, then you usually have good reason to suspect that the residuals also will be non-normally distributed.

An excellent way to visually compare a data distribution to a normal distribution is to superimpose a histogram of the data and a normal curve. To do so, we must proceed in two steps:

1. tell R that we want to make a histogram with a density curve superimposed
2. tell R that we want this to be done for both locations.

- Using the data file `sturgeon.csv` , generate histograms for `fklngth` data at The Pas and Cumberland House.

```
# use "sturgeon" dataframe to make plot called mygraph
# and define x axis as representing fklngth
mygraph <- ggplot(
  data = sturgeon,
  aes(x = fklngth)
) +
  xlab("Fork length (cm)")
# add data to the mygraph ggplot
mygraph <- mygraph +
  geom_density() + # add data density smooth
  geom_rug() + # add rug (bars at the bottom of the plot)
  geom_histogram( # add black semitransparent histogram
    aes(y = ..density..),
    color = "black", alpha = 0.3
  ) +
  # add normal curve in red, with mean and sd from fklength
  stat_function(
    fun = dnorm,
    args = list(
      mean = mean(sturgeon$fklngth),
      sd = sd(sturgeon$fklngth)
    ),
    color = "red"
  )
# display graph, by location
mygraph + facet_grid(. ~ location)
```

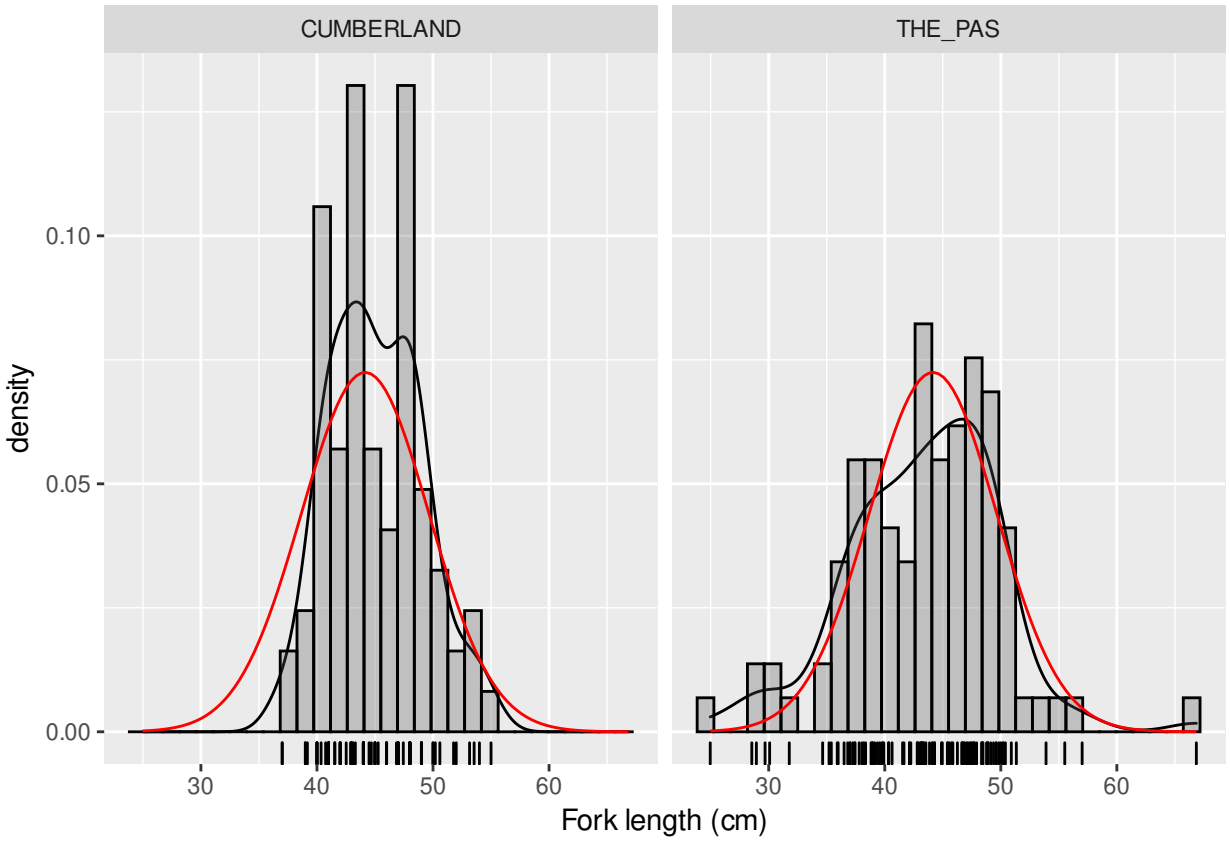


Figure 4.1: Distribution of sturgeon length at 2 locations

Based on your visual inspection, are the two samples normally distributed? Visual inspection of these plots suggests that this variable is approximately normally distributed in each sample.

Since we are interested in finding out if mean fish size differs among the two locations, it is probably also a good idea to generate a graph that compares the two groups of data. A box plot works well for this.

- Generate a box plot of `fklngth` grouped by `location` . What do you conclude about differences in size among the two locations?

```
ggplot(data = sturgeon, aes(
  x = location,
  y = fklngth
)) +
  geom_boxplot(notch = TRUE)
```

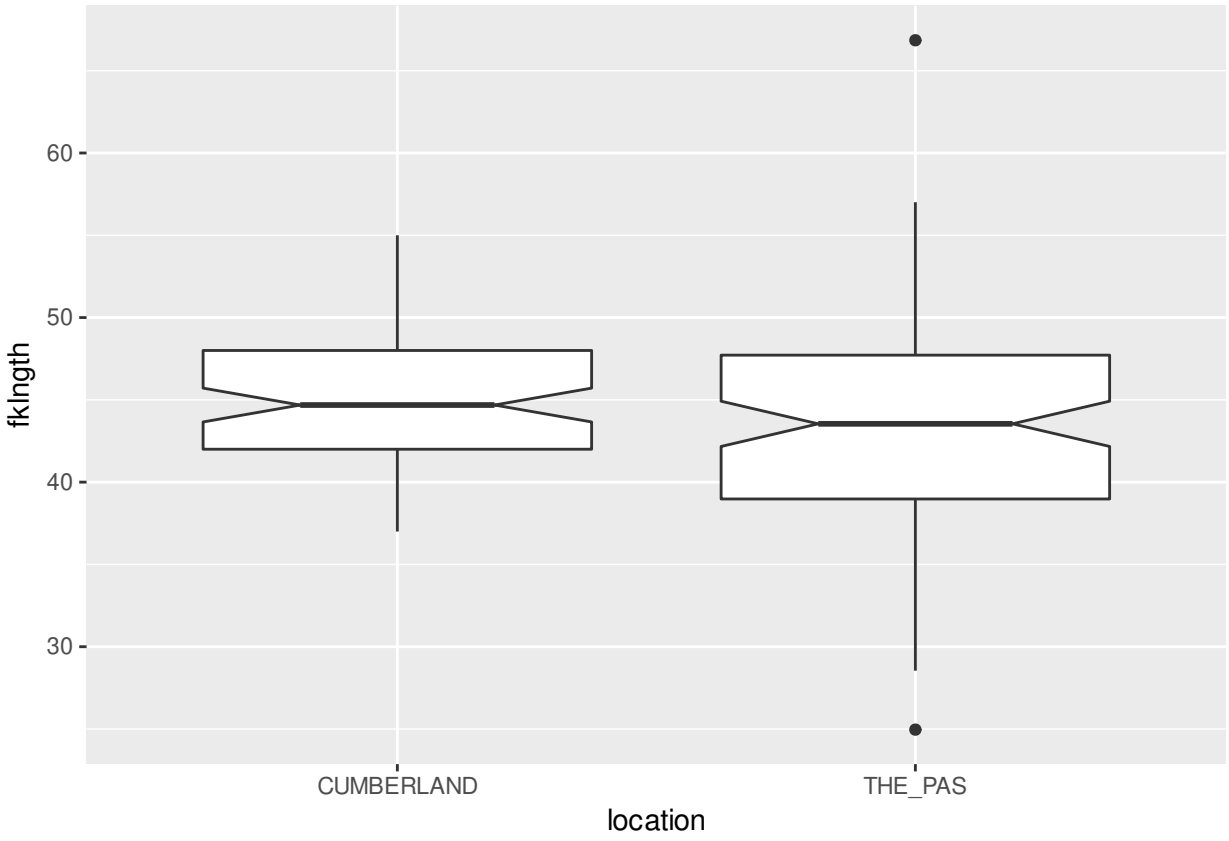


Figure 4.2: Boxplot of sturgeon legnth at 2 locations

It would appear as though there are not big differences in fish size among the two locations, although fish size at The Pas looks to be more variable, with a bigger range in size and outliers (defined as values > 1.5 * inter-quartile range) at both ends of the distribution.

4.3 Comparing means of two independent samples: parametric and non-parametric comparisons

Test the null hypothesis that the mean fklngth of The Pas and Cumberland House samples are the same. Using 3 different tests:

1. parametric test with equal variances
2. parametric test with unequal variances
3. non-parametric test

What do you conclude?

```
# t-test assuming equal variances
t.test(
  fklngth ~ location,
  data = sturgeon,
```

```
  alternative = "two.sided",
  var.equal = TRUE
)
```

```
##
##  Two Sample t-test
##
## data:  fklngth by location
## t = 2.1359, df = 184, p-value = 0.03401
## alternative hypothesis: true difference in means between group CUMBERLAND and group THE
## 95 percent confidence interval:
##  0.1308307 3.2982615
## sample estimates:
## mean in group CUMBERLAND    mean in group THE_PAS
##                 45.08439                 43.36984
```

```
# t-test assuming unequal variances
t.test(
  fklngth ~ location,
  data = sturgeon,
  alternative = "two.sided",
  var.equal = FALSE
)
```

```
##
##  Welch Two Sample t-test
##
## data:  fklngth by location
## t = 2.2201, df = 169.8, p-value = 0.02774
## alternative hypothesis: true difference in means between group CUMBERLAND and group THE
## 95 percent confidence interval:
##  0.1900117 3.2390804
## sample estimates:
## mean in group CUMBERLAND    mean in group THE_PAS
##                 45.08439                 43.36984
```

```
# test non paramétrique
wilcox.test(
  fklngth ~ location,
  data = sturgeon,
  alternative = "two.sided"
)
```

```
##
##  Wilcoxon rank sum test with continuity correction
```

```
##
## data:  fklngth by location
## W = 4973, p-value = 0.06296
## alternative hypothesis: true location shift is not equal to 0
```

Based on the *t-test*, we would reject the null hypothesis, *i.e.* there is a significant (but not highly significant) difference in mean fork length between the two populations.

Note that using the Wilcoxon rank sum test, we do not reject the null hypothesis. The two different tests therefore give us two different results. The significant difference obtained using the t-test may, at least in part, be due to deviations from normality or homoscedasticity; on the other hand, the non-significant difference obtained using the U -statistic may be due to the fact that for fixed sample size, the power of a non- parametric test is lower than the corresponding parametric test. Given the p values obtained from both tests, and the fact that for samples of this size (84 and 101), the t-test is comparatively robust with respect to non-normality, I would be inclined to reject the null hypothesis. In practice to avoid P-hacking, you should decide which test is appropriate first and then apply and interpret it, or if you decide to do all you should present results of all and interpret accordingly.

Before accepting the results of the parametric t-test and rejecting the null hypothesis that there is no difference in size between the two locations, one should do some sort of assessment to determine if the model fits the assumption of normally distributed residuals and equal variances. Preliminary examination of the raw data suggested the data appeared roughly normal but there might be problems with variances (since the spread of data for The_Pas was much greater than for Cumberland). We can examine this more closely by looking at the residuals. An easy way to do so, is to fit a linear model and use the residual diagnostic plots:

```
m1 <- lm(fklngth ~ location, data = sturgeon)
par(mfrow = c(2, 2))
plot(m1)
```

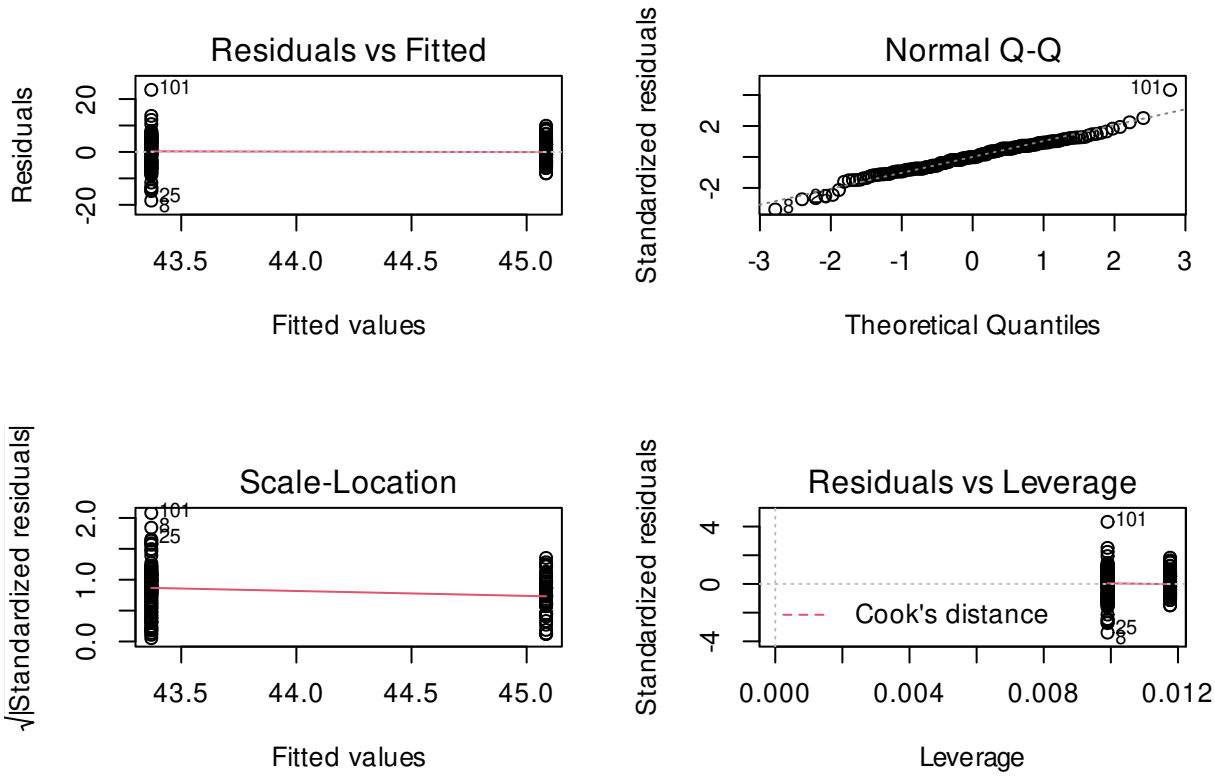


Figure 4.3: Model assumption checks

The first plot above shows the spread of the residuals around the estimated values for the two groups and allows us to get a feel for whether there are problems with the assumption of homogeneity of variances. If the variances were equal, the vertical spread of the two clusters of points should be about the same. The above plot shows that the vertical spread of the group with the smaller mean is greater than it is for the larger mean, suggesting again that there are problems with the variances. We can test this formally by examining the mean differences in the absolute value of the residuals.

The second graph above is a normal QQ plot (or probability plot) of the residuals of the model. Note that these generally fall on a straight line, suggesting there is no real problem with normality. We can do a formal test for normality on the residuals using the Shapiro-Wilk test.

```
shapiro.test(residuals(m1))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(m1)
## W = 0.97469, p-value = 0.001857
```

Hummm. The test indicates that the residuals are not normal. But, given that (a) the distribution is not very far (at least visually) from normal, and that (b) the number of observations in each

location is reasonably large (i.e. >30), we do not need to be overly concerned with this violation of the normality assumption.

How about equality of variance?

```
library(car)
leveneTest(m1)
```

```
## Warning in leveneTest.default(y = y, group = group, ...): group coerced to
## factor.
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value    Pr(>F)
## group   1  11.514 0.0008456 ***
##       184
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
bptest(m1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  m1
## BP = 8.8015, df = 1, p-value = 0.00301
```

The above are the results of two tests implemented in R (in the `car` and `lmtest` packages that can be used to test for equal variances in t-tests or linear models involving only discontinuous or categorical independent variables. **Doing the two of them is overkill**. There is not much to prefer one test over another. Levene test is possibly the better known. It tests whether the mean of absolute values of the residuals differs among groups. The Breusch-Pagan test has the advantage of being applicable to more linear models (it can deal with regression-type continuous independent variables, at least to some extent). It tests whether the studentized (i.e. scaled by their sd estimate) squared residuals vary with the independent variables in a linear model. In this case, both indicate that variances are unequal.

On the basis of these results, we conclude that there is evidence (albeit weak) to reject the null hypothesis of no difference in `fklngth` by `location`. We have modified the `t-test` to accommodate unequal variances, and are satisfied that the assumption of normally distributed residuals is sufficiently met. Thus, it appears that `fklngth` at Cumberland is greater than `fklngth` at The Pas.

4.4 Bootstrap and permutation tests to compare 2 means

4.4.1 Bootstrap

Bootstrap and permutation tests can be used to compare means (or other statistics) between pairs of samples. The general idea is simple, and it can be implemented in more ways than I can count.

Here, I use existing tools and the fact that a comparison of means can be construed as a test of a linear model. We will be able to use similar code later on when we fit more complex (but fun!) models.

```
library(boot)
```

The first section defines the function that I called bs that simply extracts coefficients from a fitted model:

```
# function to obtain model coefficients for each iteration
bs <- function(formula, data, indices) {
  d <- data[indices, ]
  fit <- lm(formula, data = d)
  return(coef(fit))
}
```

The second section with the `boot()` command is where the real work is done: take data in sturgeon, bootstrap $R = 1000$ times, each time fit the model `fklngth` vs `location`, and keep the values calculated by the `bs()` function.

```
# bootstrapping with 1000 replications
results <- boot(
  data = sturgeon, statistic = bs, R = 1000,
  formula = fklngth ~ location
)
# view results
results
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = sturgeon, statistic = bs, R = 1000, formula = fklngth ~
##     location)
##
##
## Bootstrap Statistics :
##      original      bias    std. error
## t1* 45.084391 -0.01616052   0.4249350
## t2* -1.714546  0.03934641   0.7715393
```

So we get the original estimates for the two coefficients in this model: the mean at the first (alphabetical) location, Cumberland, and the difference in means between Cumberland and The Pas). It is the second parameter, the difference between means, which is of interest here.

```
plot(results, index = 2)
```

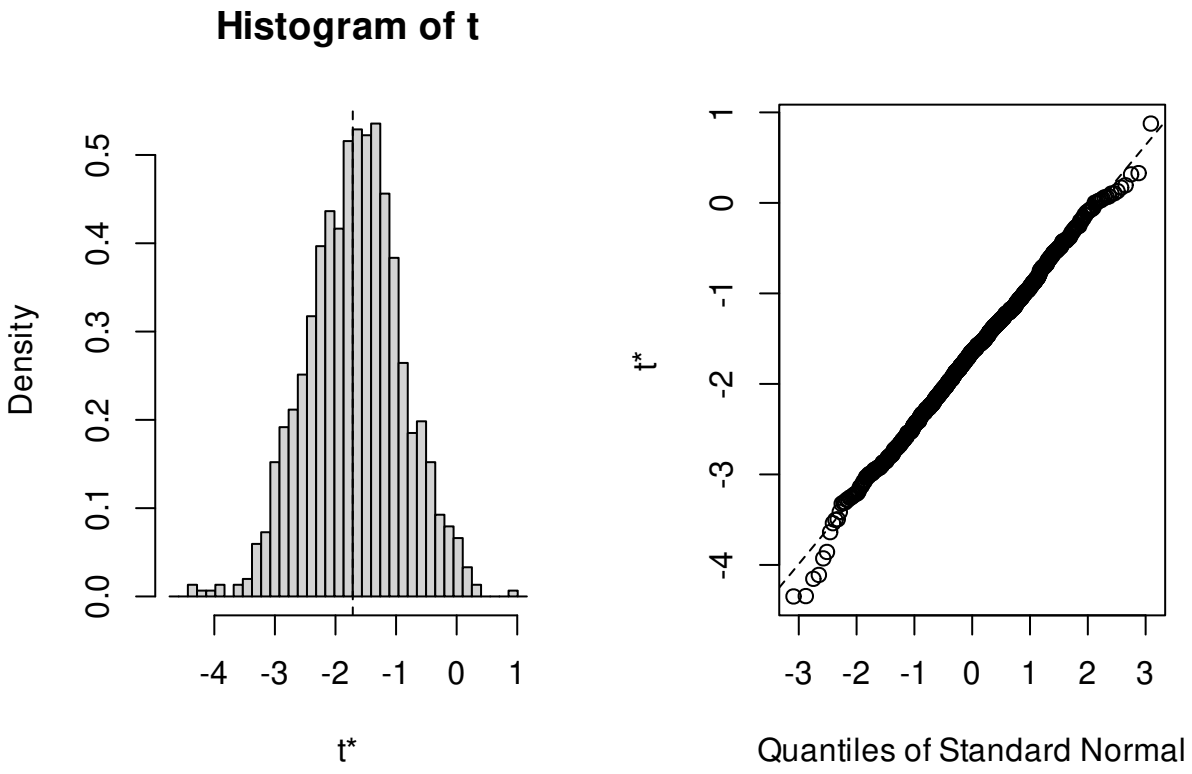


Figure 4.4: Distribution of bootstrapped mean difference

```
# get 95% confidence intervals
boot.ci(results, type = "bca", index = 2)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = results, type = "bca", index = 2)
##
## Intervals :
## Level       BCa
## 95%   (-3.261, -0.270 )
## Calculations and Intervals on Original Scale
```

The 95% CI for the difference between the two means does not include 0. Hence, the bootstrap test indicates that the two means are not equals.

4.4.2 Permutation

Permutation tests for linear models can easily be done using the `lmPerm` package .

```
m1Perm <- lmp(
  fklngth ~ location,
  data = sturgeon,
  perm = "Prob"
)
```

```
## [1] "Settings:  unique SS "
```

The `lmp()` function does all the work for us. Here it is run with the option perm to control the stopping rule used. Option Prob stops the sampling when the estimated standard deviation of the p-value falls below some fraction of the estimated. It is one of many stopping rules that one could use to do permutations on a subset of all the possibilities (because it would take foreeeever to do them all, even on your fast machine).

```
summary(m1Perm)
```

```
##
## Call:
## lmp(formula = fklngth ~ location, data = sturgeon, perm = "Prob")
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -18.40921  -3.75370  -0.08439   3.76598  23.48055
##
## Coefficients:
##           Estimate Iter Pr(Prob)
## location1   0.8573 3818   0.0257 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.454 on 184 degrees of freedom
## Multiple R-Squared: 0.02419, Adjusted R-squared: 0.01889
## F-statistic: 4.562 on 1 and 184 DF,  p-value: 0.03401
```

1. `Iter` coefficient: the Prob stopping rule stopped after 3818 iterations. Note that this number will vary each time you run this snippet of code. These are random permutation results, so expect variability.
2. `Pr(Prob)` coefficient: The estimated probability associated to H0 is 0.0257 . The observed difference in fklngth between the two locations was larger than the permuted differences in about (1 - 0.0257= about 97.4%) of the 3818 cases. Mind you, 3818 permutations is not a large number, so small p values can't be expected to be very precise. If it is critical that you get more precise p values, more permutations would be needed. Two parameters can be tweaked: maxIter, the maximum number of iterations (default=5000), and Ca, that stops

iterations when estimated standard error of the estimated p is less than Ca*p. Default 0.1.
3. `F-statistic`: The rest is the standard output for the model fitted to the data, with the standard parametric test. Here the p-value, assuming all assumptions are met, is 0.034.

4.5 Comparing the means of paired samples

In some experimental designs, individuals are measured twice: common examples are the measurement of the same individual at two different times during development, or of the same individual subjected to two different experimental treatments. In these cases, the two samples are not independent (they include the same individuals), and a paired comparison must be made.

The file `skulldat_2020.csv` shows measurements of lower face width of 15 North American girls measured at age 5 and again at age 6 years (data from Newman and Meredith, 1956).

- Let's first run a standard t-test comparing the face width at age 5 and 6, not taking into account that the data are not independent and that they are consecutive measurements on the same individuals.

```
skull <- read.csv("data/skulldat_2020.csv")
t.test(width ~ age,
  data = skull,
  alternative = "two.sided",
  paired = FALSE
)
```

```
##
##  Welch Two Sample t-test
##
## data:  width by age
## t = -1.7812, df = 27.93, p-value = 0.08576
## alternative hypothesis: true difference in means between group 5 and group 6 is not equa
## 95 percent confidence interval:
##  -0.43002624  0.03002624
## sample estimates:
## mean in group 5 mean in group 6
##        7.461333        7.661333
```

So far, we specified the t-test using a `formula` notation as `y ~ x` where `y` is the variable for which we want to compare the means and `x` is a variable defining the groups. This works really well when the samples are not paired and when the data is presented in a *long* format. For example the`skull` data is presented in a long format and contains 3 variables:

- `width`: head width for each observations
- `age`: age at measurement 5 or 6
- `id`: person identity

```r
head(skull)
```

```
##   width age id
## 1  7.33   5  1
## 2  7.53   6  1
## 3  7.49   5  2
## 4  7.70   6  2
## 5  7.27   5  3
## 6  7.46   6  3
```

When data are paired, we need to indicate how they are paired. In the `skull`data, samples are paired by an individual identity, `id`, with mearurement taken at different ages. However, the function `t.test` does not cope well with this data structure. We need to transpose the data from a *long* to a *wide* format where we have a column per group, with the data of a given individual on the same line. Here is how we can do it.

```r
skull_w <- data.frame(id = unique(skull$id))
skull_w$width5 <- skull$width[match(skull_w$id, skull$id) & skull$age == 5]
skull_w$width6 <- skull$width[match(skull_w$id, skull$id) & skull$age == 6]
head(skull_w)
```

```
##   id width5 width6
## 1  1   7.33   7.53
## 2  2   7.49   7.70
## 3  3   7.27   7.46
## 4  4   7.93   8.21
## 5  5   7.56   7.81
## 6  6   7.81   8.01
```

Now, let's run the appropriate paired t-test. What do you conclude? Compare this with the previous result and explain any differences.

```r
t.test(skull_w$width5, skull_w$width6,
  alternative = "two.sided",
  paired = TRUE
)
```

```
##
##  Paired t-test
##
## data:  skull_w$width5 and skull_w$width6
## t = -19.72, df = 14, p-value = 1.301e-11
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2217521 -0.1782479
## sample estimates:
```

```
## mean of the differences
##                    -0.2
```

The first analysis above assumes that the two samples of girls at age 5 and 6 are independent samples, whereas the second analysis assumes that the same girl is measured twice, once at age 5 and once at age 6 years.

Note that in the former case, we accept the null based on $p = 0.05$, but in the latter we reject the null. In other words, the appropriate (paired sample) test shows a very significant effect of age, whereas the inappropriate one does not. The reason is because there is a strong correlation between face width at age 5 and face width at age 6:

```
graphskull <- ggplot(data = skull_w, aes(x = width5, y = width6)) +
  geom_point() +
  labs(x = "Skull width at age 5", y = "Skull width at age 6") +
  geom_smooth() +
  scale_fill_continuous(low = "lavenderblush", high = "red")
graphskull
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

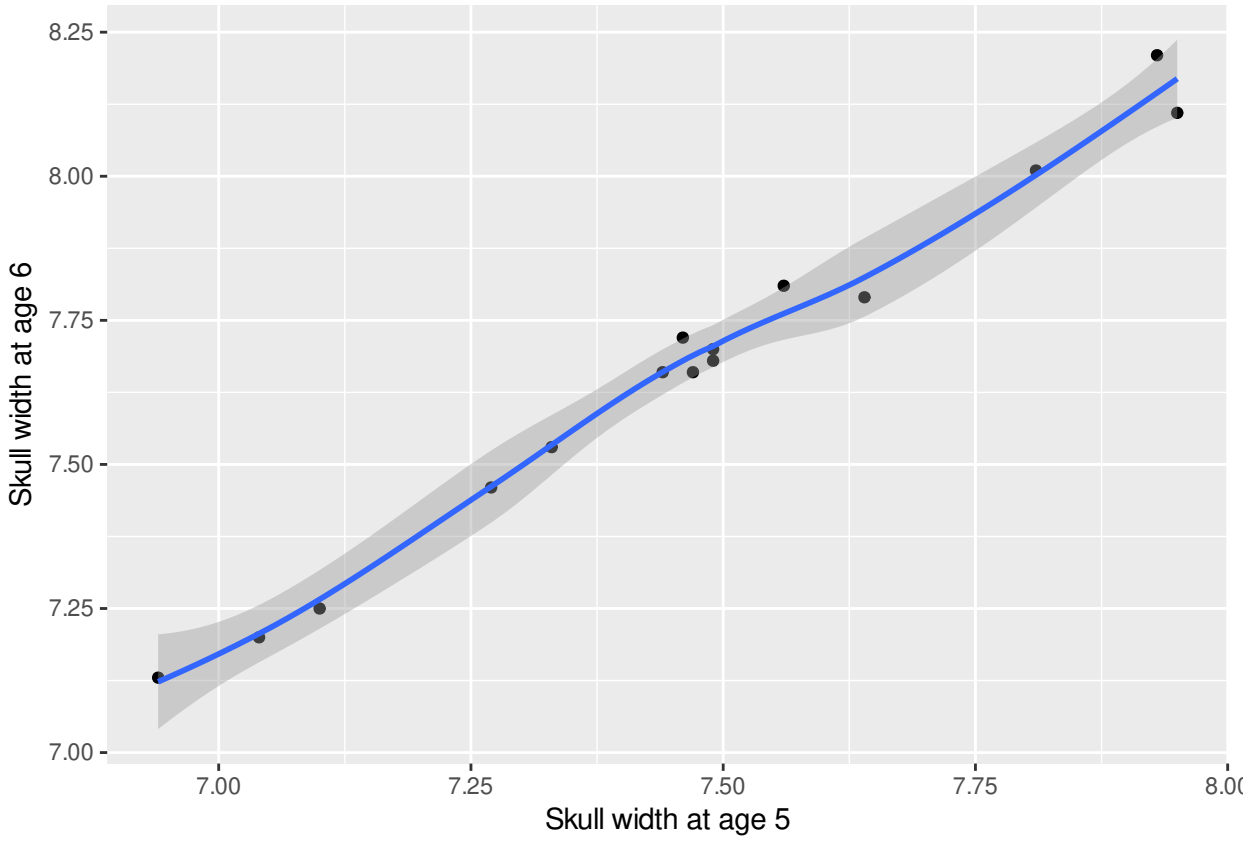


Figure 4.5: Relation between head width at age 5 and 6

With $r = 0.9930841$. In the presence of correlation, the standard error of the pairwise difference

in face width at age 5 and 6 is much smaller than the standard error of the difference between the mean face width at age 5 and 6. Thus, the associated t-statistic will be much larger for a paired sample test, *i.e.* the power of the test is much greater, and the p values are smaller.

- Repeat the above procedure with the nonparametric alternative, the Wilcoxon signed-rank test. What do you conclude?

```
wilcox.test(skull_w$width5, skull_w$width6,
  alternative = "two.sided",
  paired = TRUE
)
```

```
## Warning in wilcox.test.default(skull_w$width5, skull_w$width6, alternative =
## "two.sided", : cannot compute exact p-value with ties
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  skull_w$width5 and skull_w$width6
## V = 0, p-value = 0.0007193
## alternative hypothesis: true location shift is not equal to 0
```

So, we reach the same conclusion as we did using the paired sample t- test and conclude there are significant differences in skull sizes of girls aged 5 and 6 (what a surprise!).

But, wait a minute. We have used two-tailed tests here. But, given what we know about how children grow, a one-tail hypothesis would be preferable. This can be done by changing the alternative option. One uses the alternative hypothesis to decide if it is "less" or greater". Here, we expect that if there is an effect (i.e the alternative hypothesis), width5 will be less than width6

```
t.test(skull_w$width5, skull_w$width6,
  alternative = "less",
  paired = TRUE
)
```

```
##
##  Paired t-test
##
## data:  skull_w$width5 and skull_w$width6
## t = -19.72, df = 14, p-value = 6.507e-12
## alternative hypothesis: true difference in means is less than 0
## 95 percent confidence interval:
##        -Inf -0.1821371
## sample estimates:
## mean of the differences
##                    -0.2
```