University of Hohenheim
Santa Barbara

Improving the Management of Marine Resources through Economics and Data Science

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy

in

Slowly and Painfully Working Out the Surprisingly Obvious

by

Joshua Johnson

Committee in charge:

Professor Christopher Costello, Chair
Professor Steven Gaines
Professor Ray Hilborn
Professor Olivier Deschenes

June 2018

The Dissertation of Joshua Johnson is approved.

Professor Steven Gaines

Professor Ray Hilborn

Professor Olivier Deschenes

Professor Christopher Costello, Committee Chair

May 2018

Improving the Management of Marine Resources through Economics and Data Science

by

Joshua Johnson

To Hobbes

Acknowledgements

Thanks everyone!

Abstract

Improving the Management of Marine Resources through Economics and Data Science

by

Joshua Johnson

The data say 'meh'

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Welcome to the *R Markdown* thesis template. This template is based on (and in many places copied directly from) the UW LaTeX template, but hopefully it will provide a nicer interface for those that have never used TeX or LaTeX before. Using *R Markdown* will also allow you to easily keep track of your analyses in **R** chunks of code, with the resulting plots and output included as well. The hope is this *R Markdown* template gets you in the habit of doing reproducible research, which benefits you long-term as a researcher, but also will greatly help anyone that is trying to reproduce or build onto your results down the road.

Hopefully, you won't have much of a learning period to go through and you will reap the benefits of a nicely formatted thesis. The use of LaTeX in combination with *Markdown* is more consistent than the output of a word processor, much less prone to corruption or crashing, and the resulting file is smaller than a Word file. While you may have never had problems using Word in the past, your thesis is likely going to be at least twice as large and complex as anything you've written before, taxing Word's capabilities. After working with *Markdown* and **R** together for a few weeks, we are confident this will be

your reporting style of choice going forward.

Why use it?

R Markdown creates a simple and straightforward way to interface with the beauty of LaTeX. Packages have been written in **R** to work directly with LaTeX to produce nicely formatting tables and paragraphs. In addition to creating a user friendly interface to LaTeX, *R Markdown* also allows you to read in your data, to analyze it and to visualize it using **R** functions, and also to provide the documentation and commentary on the results of your project. Further, it allows for **R** results to be passed inline to the commentary of your results. You'll see more on this later.

Who should use it?

Anyone who needs to use data analysis, math, tables, a lot of figures, complex cross-references, or who just cares about the final appearance of their document should use *R Markdown*. Of particular use should be anyone in the sciences, but the user-friendly nature of *Markdown* and its ability to keep track of and easily include figures, automatically generate a table of contents, index, references, table of figures, etc. should make it of great benefit to nearly anyone writing a thesis project.

Chapter 2

R Markdown Basics

Here is a brief introduction into using *R Markdown*. *Markdown* is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. *R Markdown* provides the flexibility of *Markdown* with the implementation of **R** input and output. For more details on using *R Markdown* see http://rmarkdown.rstudio.com.

Be careful with your spacing in *Markdown* documents. While whitespace largely is ignored, it does at times give *Markdown* signals as to how to proceed. As a habit, try to keep everything left aligned whenever possible, especially as you type a new paragraph. In other words, there is no need to indent basic text in the Rmd document (in fact, it might cause your text to do funny things if you do).

Lists

It's easy to create a list. It can be unordered like

- Item 1

- Item 2

or it can be ordered like

1. Item 1
2. Item 2

Notice that I intentionally mislabeled Item 2 as number 4. *Markdown* automatically figures this out! You can put any numbers in the list and it will create the list. Check it out below.

To create a sublist, just indent the values a bit (at least four spaces or a tab). (Here's one case where indentation is key!)

1. Item 1
2. Item 2
3. Item 3
    - Item 3a
    - Item 3b

Line breaks

Make sure to add white space between lines if you'd like to start a new paragraph. Look at what happens below in the outputted document if you don't:

Here is the first sentence. Here is another sentence. Here is the last sentence to end the paragraph. This should be a new paragraph.

Now for the correct way:

R CHUNKS

Here is the first sentence. Here is another sentence. Here is the last sentence to end the paragraph.

This should be a new paragraph.

R chunks

When you click the **Knit** button above a document will be generated that includes both content as well as the output of any embedded **R** code chunks within the document. You can embed an **R** code chunk like this (`cars` is a built-in **R** dataset):

```
summary(cars)
```

```
     speed           dist
 Min.   : 4.0   Min.   :  2.00
 1st Qu.:12.0   1st Qu.: 26.00
 Median :15.0   Median : 36.00
 Mean   :15.4   Mean   : 42.98
 3rd Qu.:19.0   3rd Qu.: 56.00
 Max.   :25.0   Max.   :120.00
```

Inline code

If you'd like to put the results of your analysis directly into your discussion, add inline code like this:

> The `cos` of 2π is 1.

Another example would be the direct calculation of the standard deviation:

> The standard deviation of `speed` in `cars` is 5.2876444.

One last neat feature is the use of the `ifelse` conditional statement which can be used to output text depending on the result of an **R** calculation:

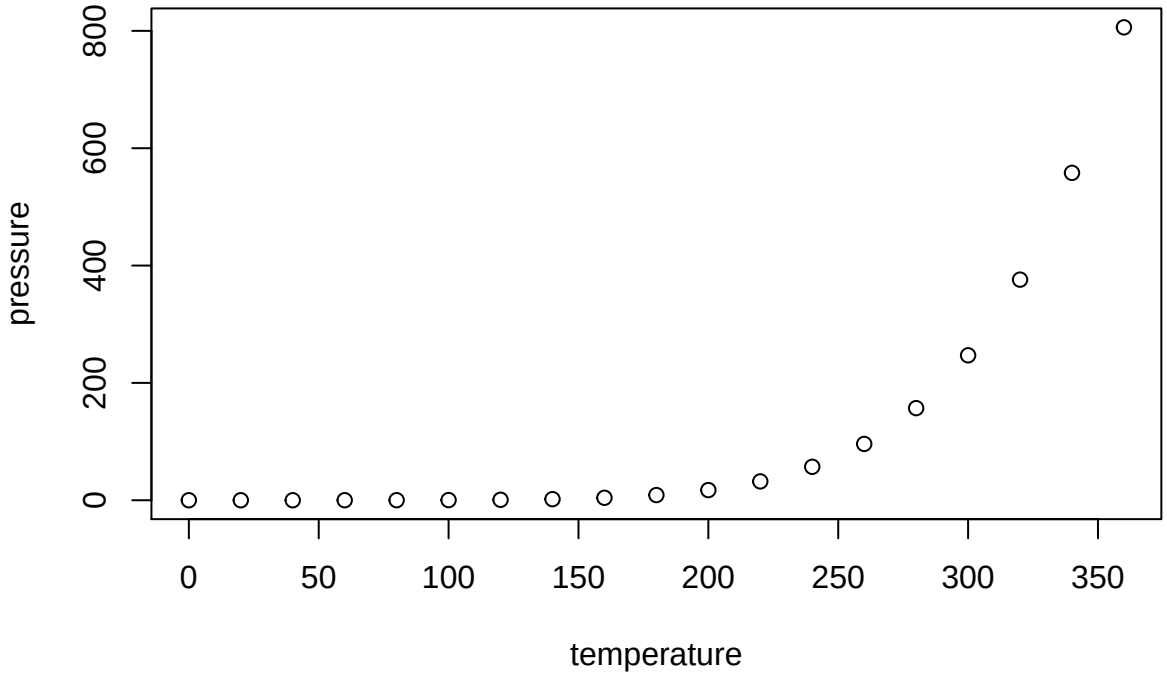
> The standard deviation is less than 6.

Note the use of `>` here, which signifies a quotation environment that will be indented.

As you see with `$2 \pi$` above, mathematics can be added by surrounding the mathematical text with dollar signs. More examples of this are in [Mathematics and Science] if you uncomment the code in Math.

Including plots

You can also embed plots. For example, here is a way to use the base **R** graphics package to produce a plot using the built-in `pressure` dataset:



Note that the `echo=FALSE` parameter was added to the code chunk to prevent printing of the **R** code that generated the plot. There are plenty of other ways to add chunk options. More information is available at http://yihui.name/knitr/options/.

Another useful chunk option is the setting of `cache=TRUE` as you see here. If document rendering becomes time consuming due to long computations or plots that are expensive to generate you can use knitr caching to improve performance. Later in this file, you'll see a way to reference plots created in **R** or external figures.

Loading and exploring data

Included in this template is a file called `flights.csv`. This file includes a subset of the larger dataset of information about all flights that departed from Seattle and Portland in 2014. More information about this dataset and its **R** package is available at http://github.com/ismayc/pnwflights14. This subset includes only Portland flights and only rows that were complete with no missing values. Merges were also done with the `airports` and `airlines` data sets in the `pnwflights14` package to get more descriptive airport and airline names.

We can load in this data set using the following command:

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

The data is now stored in the data frame called `flights` in **R**. To get a better feel for the variables included in this dataset we can use a variety of functions. Here we can see the dimensions (rows by columns) and also the names of the columns.

```
dim(flights)
```

```
[1] 52808    16
```

```
names(flights)
```

```
 [1] "month"        "day"          "dep_time"     "dep_delay"    "arr_time"
 [6] "arr_delay"    "carrier"      "tailnum"      "flight"       "dest"
[11] "air_time"     "distance"     "hour"         "minute"       "carrier_name"
[16] "dest_name"
```

LOADING AND EXPLORING DATA

Another good idea is to take a look at the dataset in table form. With this dataset having more than 50,000 rows, we won't explicitly show the results of the command here. I recommend you enter the command into the Console ***after*** you have run the **R** chunks above to load the data into **R**.

```
View(flights)
```

While not required, it is highly recommended you use the `dplyr` package to manipulate and summarize your data set as needed. It uses a syntax that is easy to understand using chaining operations. Below I've created a few examples of using `dplyr` to get information about the Portland flights in 2014. You will also see the use of the `ggplot2` package, which produces beautiful, high-quality academic visuals.

We begin by checking to ensure that needed packages are installed and then we load them into our current working environment:

```
# List of packages required for this analysis
pkg <- c("dplyr", "ggplot2", "knitr", "bookdown", "devtools")
# Check if packages are not installed and assign the
# names of the packages not installed to the variable new.pkg
new.pkg <- pkg[!(pkg %in% installed.packages())]
# If there are any packages in the list that aren't installed,
# install them
if (length(new.pkg))
  install.packages(new.pkg, repos = "http://cran.rstudio.com")
# Load packages (huskydown will load all of the packages as well)
library(gauchodown)
```

The example we show here does the following:

- Selects only the `carrier_name` and `arr_delay` from the `flights` dataset and then assigns this subset to a new variable called `flights2`.
- Using `flights2`, we determine the largest arrival delay for each of the carriers.

```
library(dplyr)
```

```
Attaching package: 'dplyr'
```

```
The following objects are masked from 'package:stats':

    filter, lag
```

```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
flights2 <- flights %>%
  select(carrier_name, arr_delay)
max_delays <- flights2 %>%
  group_by(carrier_name) %>%
  summarize(max_arr_delay = max(arr_delay, na.rm = TRUE))
```

```
`summarise()` ungrouping output (override with `.groups` argument)
```

A useful function in the `knitr` package for making nice tables in *R Markdown* is called `kable`. It is much easier to use than manually entering values into a table by copying and pasting values into Excel or LaTeX. This again goes to show how nice reproducible documents can be! (Note the use of `results="asis"`, which will produce the table instead of the code to create the table.) The `caption.short` argument is used to include a shorter title to appear in the List of Tables.

```
library(knitr)
kable(max_delays,
      col.names = c("Airline", "Max Arrival Delay"),
      caption = "Maximum Delays by Airline",
      caption.short = "Max Delays by Airline",
      longtable = TRUE,
      booktabs = TRUE)
```

Table 2.1: Maximum Delays by Airline

Airline	Max Arrival Delay
Alaska Airlines Inc.	338
American Airlines Inc.	1539
Delta Air Lines Inc.	651
Frontier Airlines Inc.	575
Hawaiian Airlines Inc.	407
JetBlue Airways	273
SkyWest Airlines Inc.	421
Southwest Airlines Co.	694

United Air Lines Inc.	472
US Airways Inc.	347
Virgin America	366

The last two options make the table a little easier-to-read.

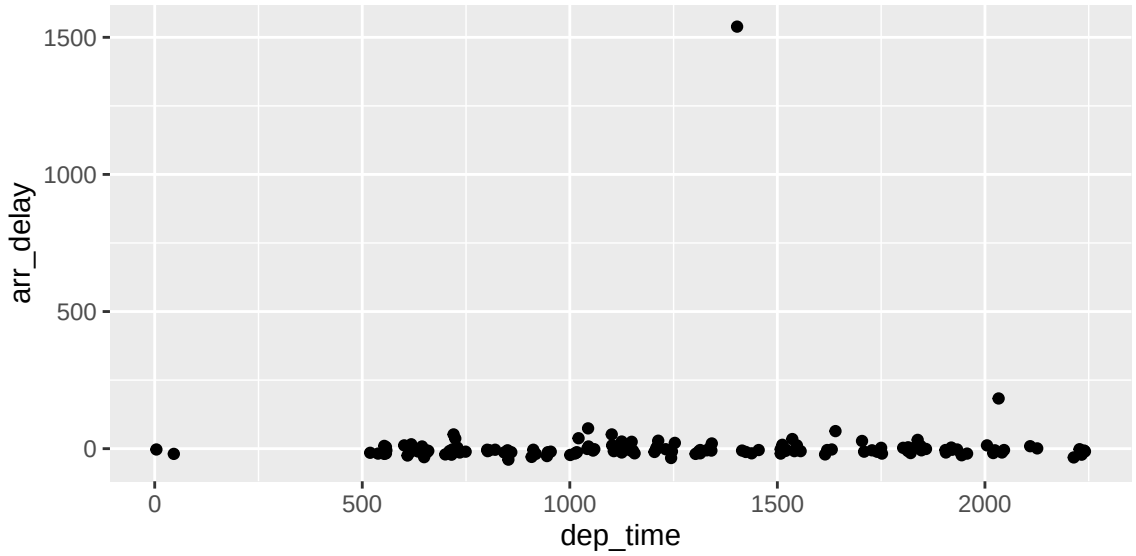
We can further look into the properties of the largest value here for American Airlines Inc. To do so, we can isolate the row corresponding to the arrival delay of 1539 minutes for American in our original `flights` dataset.

```
flights %>%
  filter(arr_delay == 1539,
         carrier_name == "American Airlines Inc.") %>%
  select(-c(month, day, carrier, dest_name, hour,
            minute, carrier_name, arr_delay))

  dep_time dep_delay arr_time tailnum flight dest air_time distance
1     1403      1553     1934  N595AA   1568  DFW      182     1616
```

We see that the flight occurred on March 3rd and departed a little after 2 PM on its way to Dallas/Fort Worth. Lastly, we show how we can visualize the arrival delay of all departing flights from Portland on March 3rd against time of departure.

```
library(ggplot2)
flights %>%
  filter(month == 3, day == 3) %>%
  ggplot(aes(x = dep_time,
```

```
            y = arr_delay)) +
  geom_point()
```



Additional resources

- *Markdown* Cheatsheet - https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet
- *R Markdown* Reference Guide - https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf
- Introduction to `dplyr` - https://cran.rstudio.com/web/packages/dplyr/vignettes/introduction.html
- `ggplot2` Documentation - http://docs.ggplot2.org/current/

Chapter 3

Results

Chapter 4

Packages, data loading etc.

```
library(tidyverse)
library(dplyr)
library(ggplot2)
library(cowplot)
library(reshape2)
library(rstatix)
library(ggpubr)
library(nlme)
library(lme4)
```

Chapter 5

Basic statistics and data mangling

```
# A tibble: 80 x 5
   type  id        n avg.qty sd.qty
   <chr> <fct> <int>   <dbl>  <dbl>
 1 rhizo 198       3 230588. 12686.
 2 rhizo 201       3 228907.  6576.
 3 rhizo 206       3 332463.  6923.
 4 rhizo 212       3 210821. 25403.
 5 rhizo 216       3 488172. 25382.
 6 rhizo 224       3 252674. 30488.
 7 rhizo 226       3 353033. 31415.
 8 rhizo 232       3 583033. 12524.
 9 rhizo 234       3 522109. 92493.
10 rhizo 237       3 287493.  3463.
# ... with 70 more rows

# A tibble: 80 x 10
```

```
   type  id        n avg.qty sd.qty dna_conc unit      qty_undil   qty_ul qty_ng
   <chr> <fct> <int>   <dbl>  <dbl>    <dbl> <chr>         <dbl>    <dbl>  <dbl>
 1 rhizo 198       3 230588. 12686.     35.8 DNA[ng_~  92235246.   1.15e7 3.22e5
 2 rhizo 201       3 228907.  6576.     38.1 DNA[ng_~  91562808.   1.14e7 3.00e5
 3 rhizo 206       3 332463.  6923.     47.5 DNA[ng_~ 132985071.   1.66e7 3.50e5
 4 rhizo 212       3 210821. 25403.     35.1 DNA[ng_~  84328527.   1.05e7 3.01e5
 5 rhizo 216       3 488172. 25382.     52.8 DNA[ng_~ 195268783.   2.44e7 4.62e5
 6 rhizo 224       3 252674. 30488.     38.6 DNA[ng_~ 101069454.   1.26e7 3.27e5
 7 rhizo 226       3 353033. 31415.     50.3 DNA[ng_~ 141213354.   1.77e7 3.51e5
 8 rhizo 232       3 583033. 12524.     60.2 DNA[ng_~ 233213183.   2.92e7 4.85e5
 9 rhizo 234       3 522109. 92493.     61.1 DNA[ng_~ 208843525    2.61e7 4.27e5
10 rhizo 237       3 287493.  3463.     43.1 DNA[ng_~ 114997088.   1.44e7 3.34e5
# ... with 70 more rows


# A tibble: 80 x 11
   type  id     qty_ng unit      treatment fert  fung  block row   time  subject
   <chr> <fct>   <dbl> <chr>     <chr>     <fct> <fct> <fct> <fct> <fct>   <dbl>
 1 rhizo 198   321871. GeneCopi~ H00       no    no    b     1     t2          1
 2 rhizo 201   300167. GeneCopi~ H00       no    no    c     1     t2          2
 3 rhizo 206   350108. GeneCopi~ H01       yes   no    b     1     t2          6
 4 rhizo 212   300572. GeneCopi~ HP0       no    yes   b     1     t2         11
 5 rhizo 216   462284. GeneCopi~ HP1       yes   yes   b     1     t2         16
 6 rhizo 224   327467. GeneCopi~ HP0       no    yes   c     2     t2         12
 7 rhizo 226   350719. GeneCopi~ HP0       no    yes   e     2     t2         13
 8 rhizo 232   484569. GeneCopi~ H00       no    no    d     2     t2          3
 9 rhizo 234   427258. GeneCopi~ HP1       yes   yes   c     2     t2         17
```

```
10 rhizo 237   333828. GeneCopi~ H01       yes   no    c     2     t2          7
# ... with 70 more rows
```

The data preparation is finished for the 16s rRNA gene copy data. We have worked with the raw quantity values used qPCR machine (standardized on the individual runs standard curve), did summary statistics (averaging the technical replicates) and calculated gene copy number per nanogram DNA.

Now we move on to the statistical modelling and checking for significance.

Chapter 6

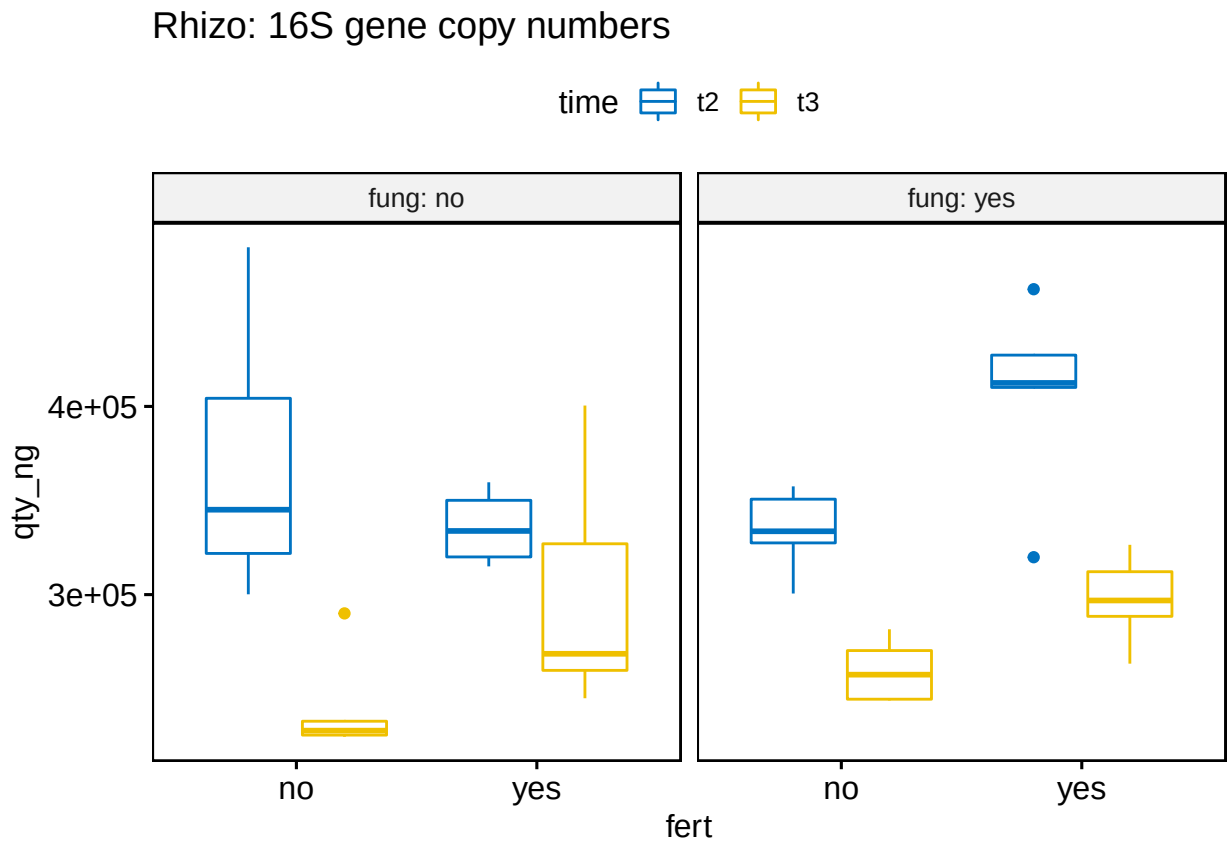
Statistical modelling and sig. testing

Rhizo

Checking data balance

```
# A tibble: 8 x 7
  fert  fung  time  variable     n    mean     sd
  <fct> <fct> <fct> <chr>    <dbl>   <dbl>  <dbl>
1 no    no    t2    qty_ng       5 371209. 74345.
2 no    no    t3    qty_ng       5 240046. 28114.
3 no    yes   t2    qty_ng       5 333985. 22320.
4 no    yes   t3    qty_ng       5 259493. 16454.
5 yes   no    t2    qty_ng       5 335729. 19108.
6 yes   no    t3    qty_ng       5 300139. 64159.
7 yes   yes   t2    qty_ng       5 406397. 52670.
8 yes   yes   t3    qty_ng       5 297449. 24004.
```

We have a balanced dataset (n is the same for each group). Moving on to a boxplot to visually inspect groups and look for outliers.



We have one outlier in [fert:no|fung:no|time:t3] and two outliers in [fert:yes|fung:yes|time:t2]. Identifying the outliers:

```
# A tibble: 3 x 10
  fert  fung  time  id     qty_ng block row   subject is.outlier is.extreme
  <fct> <fct> <fct> <fct>   <dbl> <fct> <fct>   <dbl> <lgl>      <lgl>
1 no    no    t3    355   290010. e     3           4 TRUE       TRUE
2 yes   yes   t2    216   462284. b     1          16 TRUE       FALSE
3 yes   yes   t2    253   319869. f     3          18 TRUE       TRUE
```

Replicate 216, 355 and 253 are outliers. The two last ones are extreme outliers. Consult

the 'index' file to see the differences between the replicates and their respective groups.

> Values above Q3 + 1.5xIQR or below Q1 - 1.5xIQR are considered as outliers.
> Values above Q3 + 3xIQR or below Q1 - 3xIQR are considered as extreme points (or extreme outliers).

We will perform the following analysis with and without the extreme outliers.

Normality and heteroskedasticity

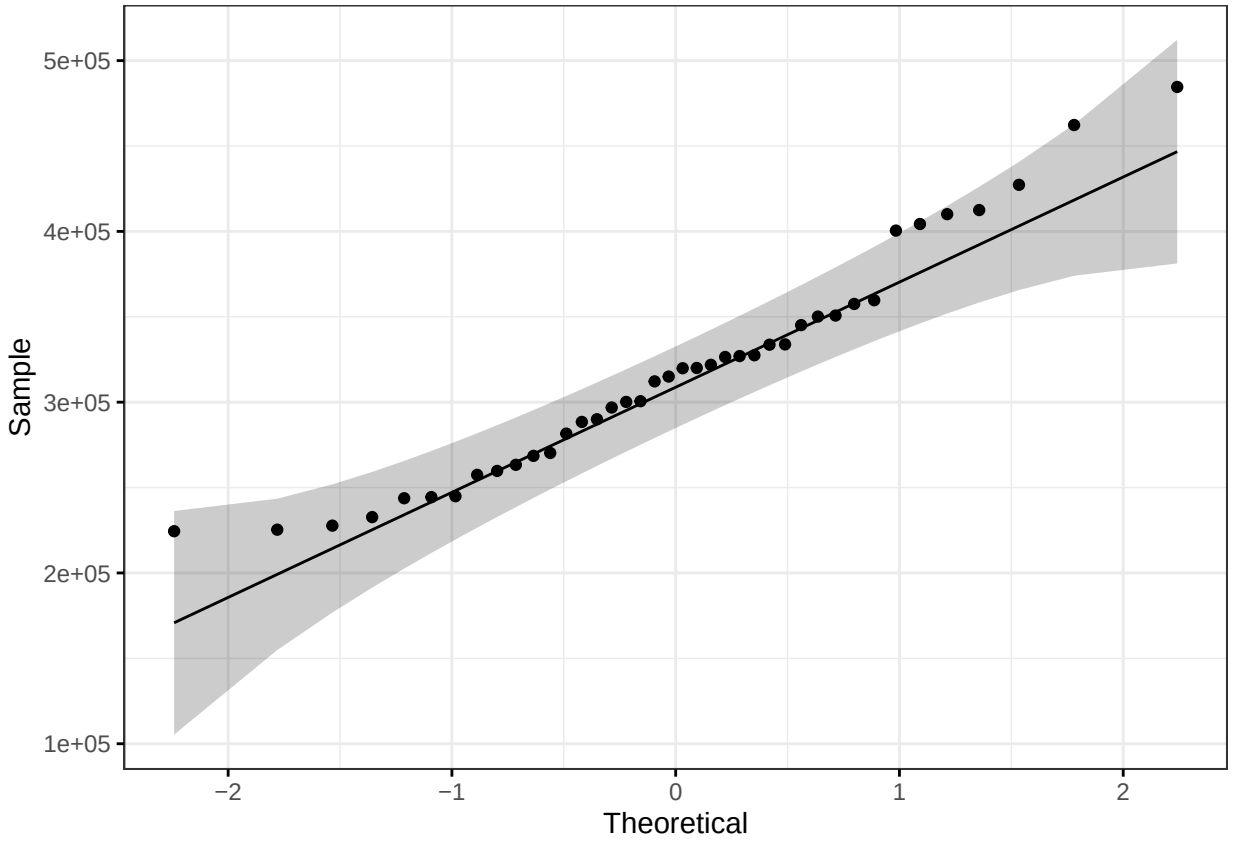
```
# Normality test with all outliers (Shapiro Test)
```

```
genecopy.work %>%
  group_by(fert, fung, time) %>%
  shapiro_test(qty_ng)
```

```
# A tibble: 8 x 6
  fert  fung  time  variable statistic       p
  <fct> <fct> <fct> <chr>        <dbl>   <dbl>
1 no    no    t2    qty_ng       0.918 0.517
2 no    no    t3    qty_ng       0.651 0.00271
3 no    yes   t2    qty_ng       0.948 0.721
4 no    yes   t3    qty_ng       0.909 0.464
5 yes   no    t2    qty_ng       0.935 0.633
6 yes   no    t3    qty_ng       0.869 0.262
7 yes   yes   t2    qty_ng       0.880 0.308
8 yes   yes   t3    qty_ng       0.987 0.968
```

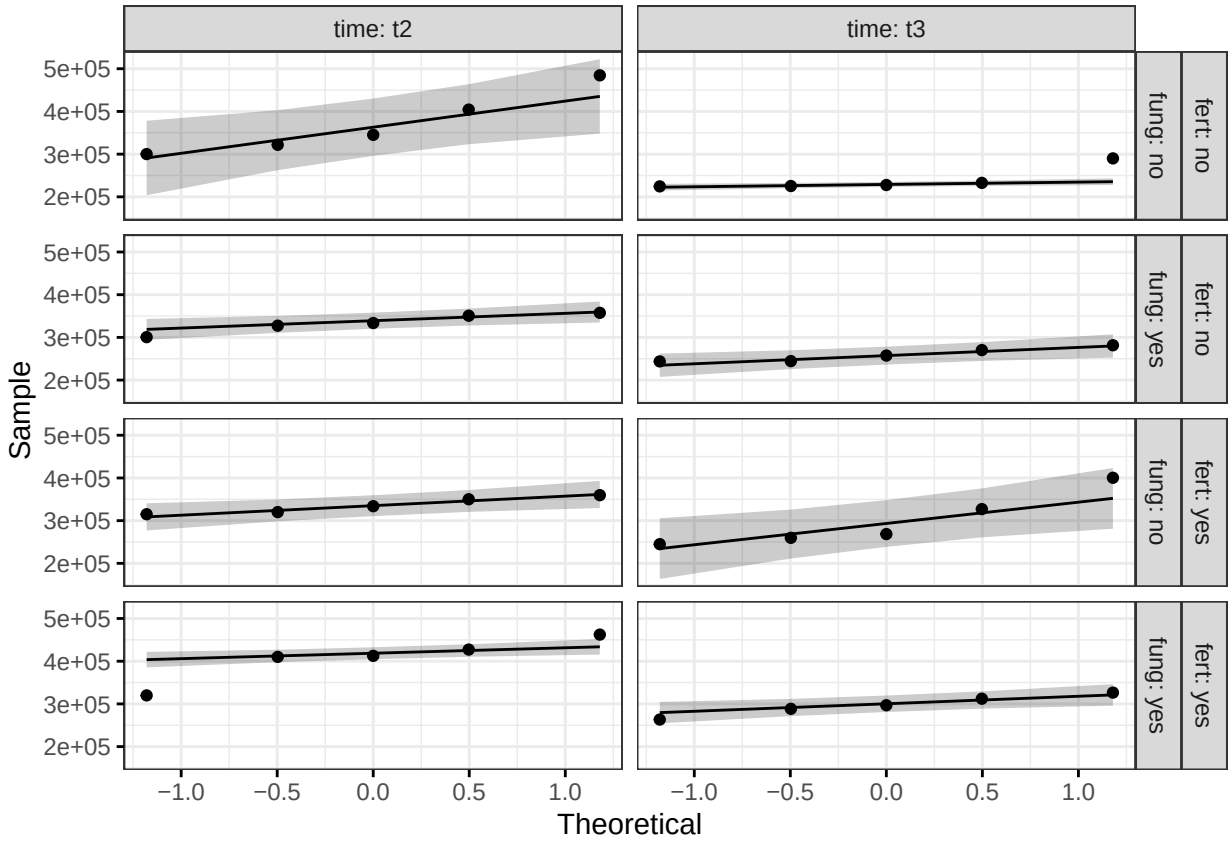
```
# QQplots: Investigate heteroskedasticity visually
```

```
  # All datapoints combined
   genecopy.work %>%
    ggqqplot("qty_ng", ggtheme = theme_bw())
```




```
  # Grouped by treatments and timepoints
   genecopy.work %>%
    ggqqplot("qty_ng", ggtheme = theme_bw())+
    facet_grid(fert + fung ~ time, labeller = "label_both")
```

RHIZO




```
# The deviation in [fert:no|fung:no|time:t3] is clearly visible, exclude the outlie
  # [fert:yes|fung:yes|time:t2] shows also an outlier far away from the line, somew
  # Repeat normality test with filter of outliers
  # Heteroskedascticity not perfect but OK
```

Except for one group, all respective p-values above 0.05. The treatment group with a p-value under 0.05 had an outlier (see outlier check).

QQplots look fine.

Let's redo the normality (shapiro) test without the outlier in the group which failed the normality test.

```
# Shapiro Test with outliers removed
```

```
  genecopy.work %>%
    filter(id != "355")%>%
    #filter(id != "253")%>%
    group_by(fert, fung, time) %>%
    shapiro_test(qty_ng)
```

```
# A tibble: 8 x 6
  fert  fung  time  variable statistic     p
  <fct> <fct> <fct> <chr>        <dbl> <dbl>
1 no    no    t2    qty_ng       0.918 0.517
2 no    no    t3    qty_ng       0.896 0.413
3 no    yes   t2    qty_ng       0.948 0.721
4 no    yes   t3    qty_ng       0.909 0.464
5 yes   no    t2    qty_ng       0.935 0.633
6 yes   no    t3    qty_ng       0.869 0.262
7 yes   yes   t2    qty_ng       0.880 0.308
8 yes   yes   t3    qty_ng       0.987 0.968
```

```
  # Now all trt grps pass the shapiro test (p > 0.05), normality can be assumed
  # QQplots look better, better heteroskedasticity
  # Let's provide a copydataset with the outliers excluded
```

```
   genecopy.work.filtered <- genecopy.work %>%
    #filter(id != "253")%>%
     filter(id != "355")
```

RHIZO

```
# Finished the preliminaries for the anova test
```

Lets move on to the ANOVA test for rhizo.

ANOVA: choosing a model

An analysis of the dataset structure is needed to find the right statistical model. The data was generated in a randomized complete block design (RCBD).

Model parameters

Fixed effects

- Fertilizer (fert) (Binary variable (yes/no))
- Fungicide and growth (fung) (Binary variable (yes/no))

Random effects

- Time
- Block
- (Row)
- (Could possibly include rainfall 3-7 days before sampling)

Mixed effect model

The model needs to account for the above listed fixed and random effects. The regular lm() function of R stats package will fit all variables as fixed effects if they are integrated

into the forumlae. Therefore we need the package nlme which can account for random effects. Because we are using only two timepoints I will stick to a linear model. Generally, I'd consider a non-linear model if all timepoints would be in the analysis. We are investigating gene copy numbers which are directly correlated and have a causal relation ship with number of bacteria. Bacteria growth is better estimated with a logistic regression.

I'll do a stepwise modelling approach without the mathematic forumlae (will be done in the thesis tho).

All main effects + Interaction (Full model)

```
# Fitting a linear mixed effect model to our genecopynumber dataset
  # Treatment variables (fert|fung) are treated as fixed effects
  # Time and Block are random effect variables (can't be replicated)
  # We use the package nlme to effectively input random effect variables for our mo
  # Interaction of fert*fung must be tested, because we have two factorial experime
  # It is also possible to just type in fert*fung as predictor variable, the packag
  # If more timepoints of this are used, a non-linear model would be better because

  fit.all <- lme(qty_ng ~ fert+fung+fert*fung,
                 random = list(~1|block, ~1|time, ~1|row),
                 data=genecopy.work.filtered)
  summary(fit.all)
```

```
Linear mixed-effects model fit by REML
  Data: genecopy.work.filtered
       AIC      BIC    logLik
```

RHIZO

```
  891.8874 904.3302 -437.9437


Random effects:
 Formula: ~1 | block
        (Intercept)
StdDev:     2.11687

 Formula: ~1 | time %in% block
        (Intercept)
StdDev:    44341.31

 Formula: ~1 | row %in% time %in% block
        (Intercept) Residual
StdDev:    23.07462 47814.73

Fixed effects:  qty_ng ~ fert + fung + fert * fung
                    Value Std.Error DF   t-value p-value
(Intercept)     303825.58  21371.60 17 14.216324  0.0000
fertyes          14108.35  22107.80 17  0.638162  0.5319
fungyes          -7086.54  22107.80 17 -0.320545  0.7525
fertyes:fungyes  41075.50  30757.19 17  1.335476  0.1993
 Correlation:
                (Intr) fertys fungys
fertyes         -0.551
fungyes         -0.551  0.532
fertyes:fungyes  0.396 -0.719 -0.719
```

```
Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-1.3954961 -0.5807666 -0.1468039  0.3914567  2.5478950

Number of Observations: 39
Number of Groups:
                    block          time %in% block row %in% time %in% block
                        5                        10                        19
```

The interaction term is not significant, so I'm droping it from the model, leaving only the main effects in.

Both main effects (Reduced model)

```
  fit.nointeraction <- lme(qty_ng ~ fert+fung,
                  random = list(~1|block, ~1|time),
                  data=genecopy.work.filtered)
  summary(fit.nointeraction)
```

```
Linear mixed-effects model fit by REML
  Data: genecopy.work.filtered
       AIC      BIC    logLik
  912.1677 921.6688 -450.0838

Random effects:
 Formula: ~1 | block
```

RHIZO

```
        (Intercept)
StdDev:    5.363311

 Formula: ~1 | time %in% block
        (Intercept) Residual
StdDev:    44539.61 48404.24

Fixed effects:  qty_ng ~ fert + fung
                Value Std.Error DF   t-value p-value
(Intercept) 292542.79  19789.96 27 14.782382  0.0000
fertyes      35322.05  15559.36 27  2.270148  0.0314
fungyes      14127.16  15559.36 27  0.907953  0.3719
 Correlation:
        (Intr) fertys
fertyes -0.418
fungyes -0.418  0.032

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-1.3802533 -0.4939084 -0.1678409  0.2697620  2.7488597

Number of Observations: 39
Number of Groups:
          block time %in% block
              5              10
```

Fertilizer (fert) is sig. but fungicide and growth regulators (fung) not. I'm dropping fung as a effect from the model.

Only Fert as main effect

```
  fit.nofung <- lme(qty_ng ~ fert,
                    random = list(~1|block,~1|time),
                    data=genecopy.work.filtered)
  summary(fit.nofung)
```

```
Linear mixed-effects model fit by REML
  Data: genecopy.work.filtered
       AIC      BIC    logLik
  932.1343 940.1888 -461.0671


Random effects:
 Formula: ~1 | block
        (Intercept)
StdDev:    5.905738

 Formula: ~1 | time %in% block
        (Intercept) Residual
StdDev:    44307.45 48320.63

Fixed effects:  qty_ng ~ fert
                Value Std.Error DF   t-value p-value
(Intercept) 300064.66  17904.63 28 16.759051  0.0000
```

SOIL

```
fertyes      34863.75  15524.34 28  2.245748  0.0328
 Correlation:
        (Intr)
fertyes -0.447

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.52321817 -0.63607437 -0.04244137  0.36656202  2.60355504

Number of Observations: 39
Number of Groups:
          block time %in% block
              5              10
```

This is the final model. We have a significant effect of fert on 16S gene copy numbers in the rhizo type soil samples.

Soil

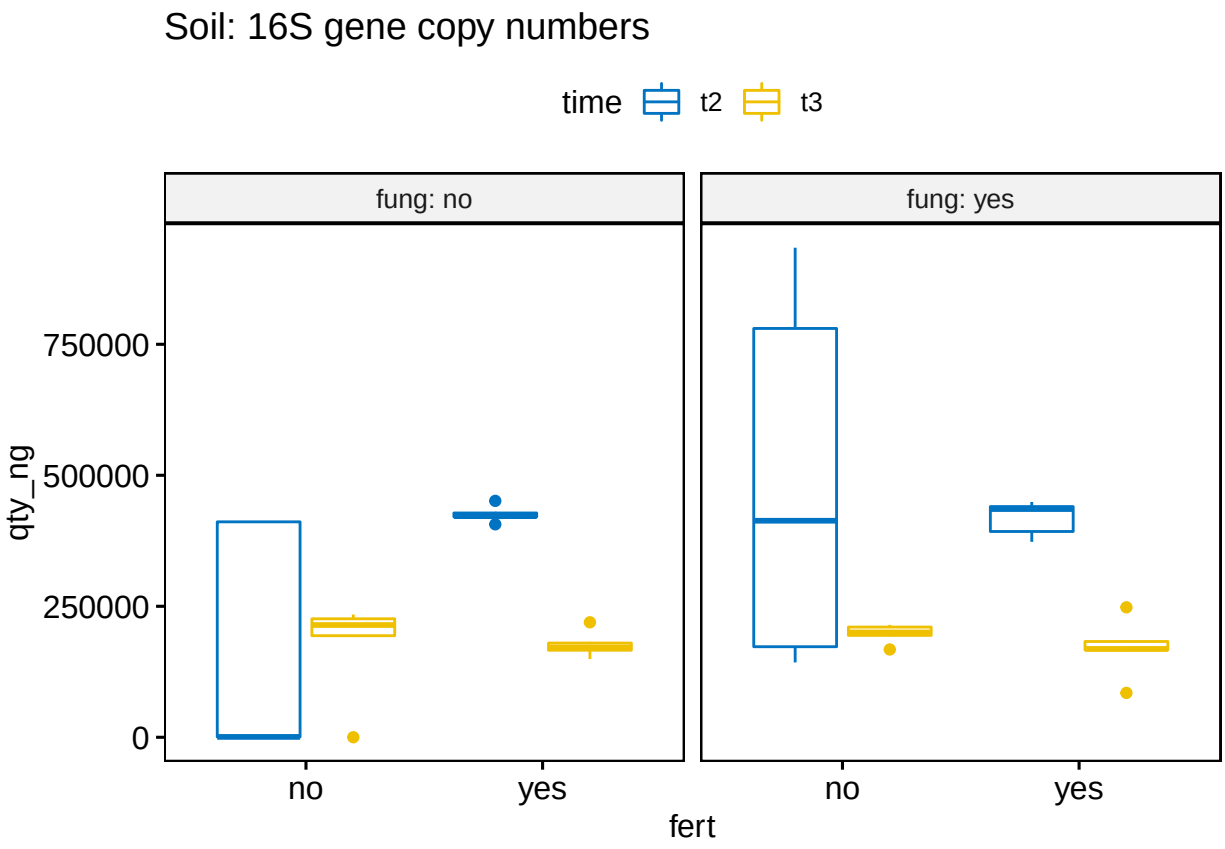
Checking the datas balance

```
# A tibble: 8 x 7
  fert  fung  time  variable     n    mean      sd
  <fct> <fct> <fct> <chr>    <dbl>   <dbl>   <dbl>
1 no    no    t2    qty_ng       5 165023. 225148.
2 no    no    t3    qty_ng       5 173726.  98246.
```

```
3 no    yes   t2    qty_ng       5 488698. 356576.
4 no    yes   t3    qty_ng       5 197417.  18488.
5 yes   no    t2    qty_ng       5 425696.  16304.
6 yes   no    t3    qty_ng       5 177266.  26063.
7 yes   yes   t2    qty_ng       5 418116.  33256.
8 yes   yes   t3    qty_ng       5 170149.  58194.
```

We have a balanced dataset (n is the same for each group). Moving on to a boxplot to visually inspect groups and look for outliers.

Boxplot to inspect data visually



SOIL

We have 7 outliers. 7 of 20 datapoints are outliers. Not good. The boxplots are very narrow for t3 and for the [fert:yes] groups.

Identifying the outliers.

```
  outlier <-  genecopy.work %>%
    group_by(fert,fung, time) %>%
    identify_outliers(qty_ng)
  outlier
```

```
# A tibble: 7 x 10
  fert  fung  time  id      qty_ng block row   subject is.outlier is.extreme
  <fct> <fct> <fct> <fct>    <dbl> <fct> <fct>   <dbl> <lgl>      <lgl>
1 no    no    t3    382       99.0 f     4           5 TRUE       TRUE
2 no    yes   t3    359   167569.  e     3          14 TRUE       FALSE
3 yes   no    t2    237   451180.  c     2           7 TRUE       TRUE
4 yes   no    t2    251   406456.  f     3           9 TRUE       FALSE
5 yes   no    t3    343   219417.  d     3           8 TRUE       FALSE
6 yes   yes   t3    365   248032.  e     4          19 TRUE       TRUE
7 yes   yes   t3    372    84678.  f     4          20 TRUE       TRUE
```

4/7 outliers are extreme:

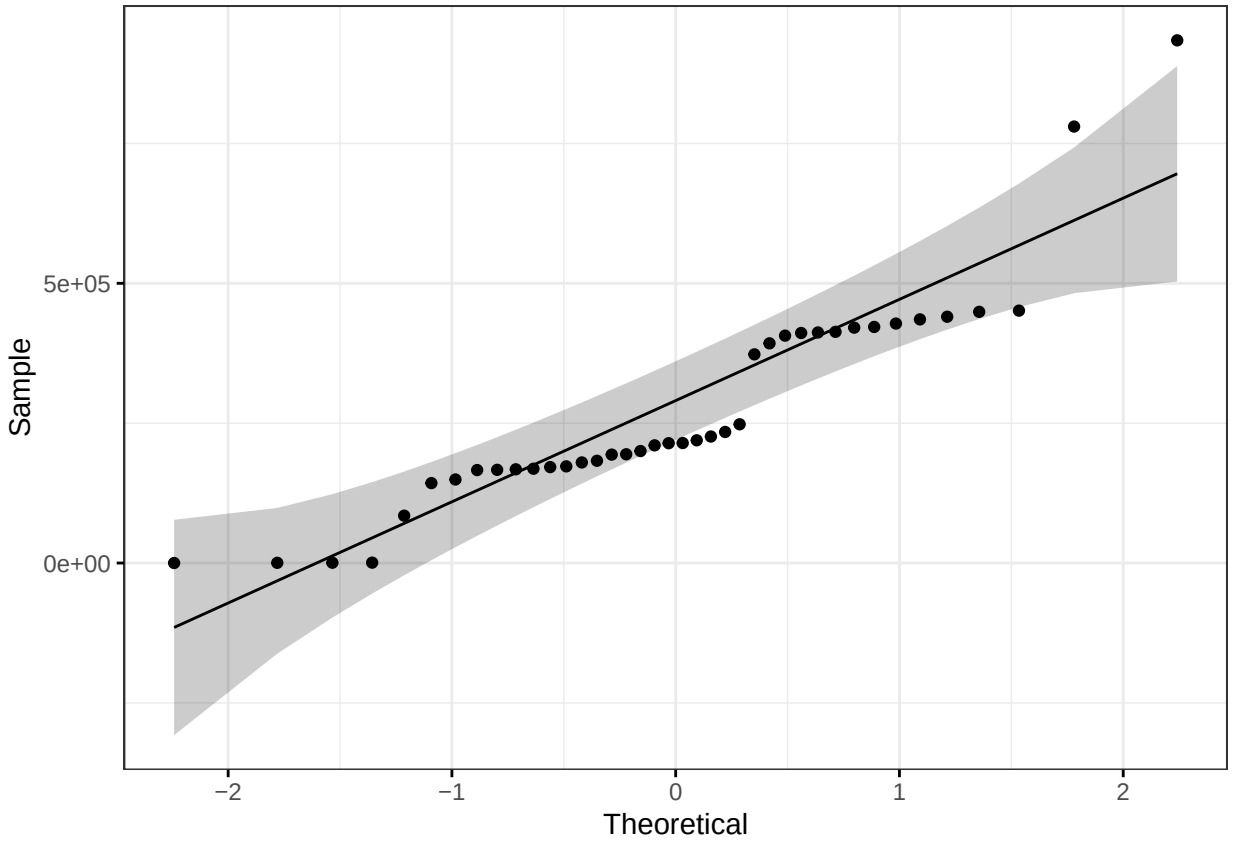
> Values above Q3 + 1.5xIQR or below Q1 - 1.5xIQR are considered as outliers. Values above Q3 + 3xIQR or below Q1 - 3xIQR are considered as extreme points (or extreme outliers).

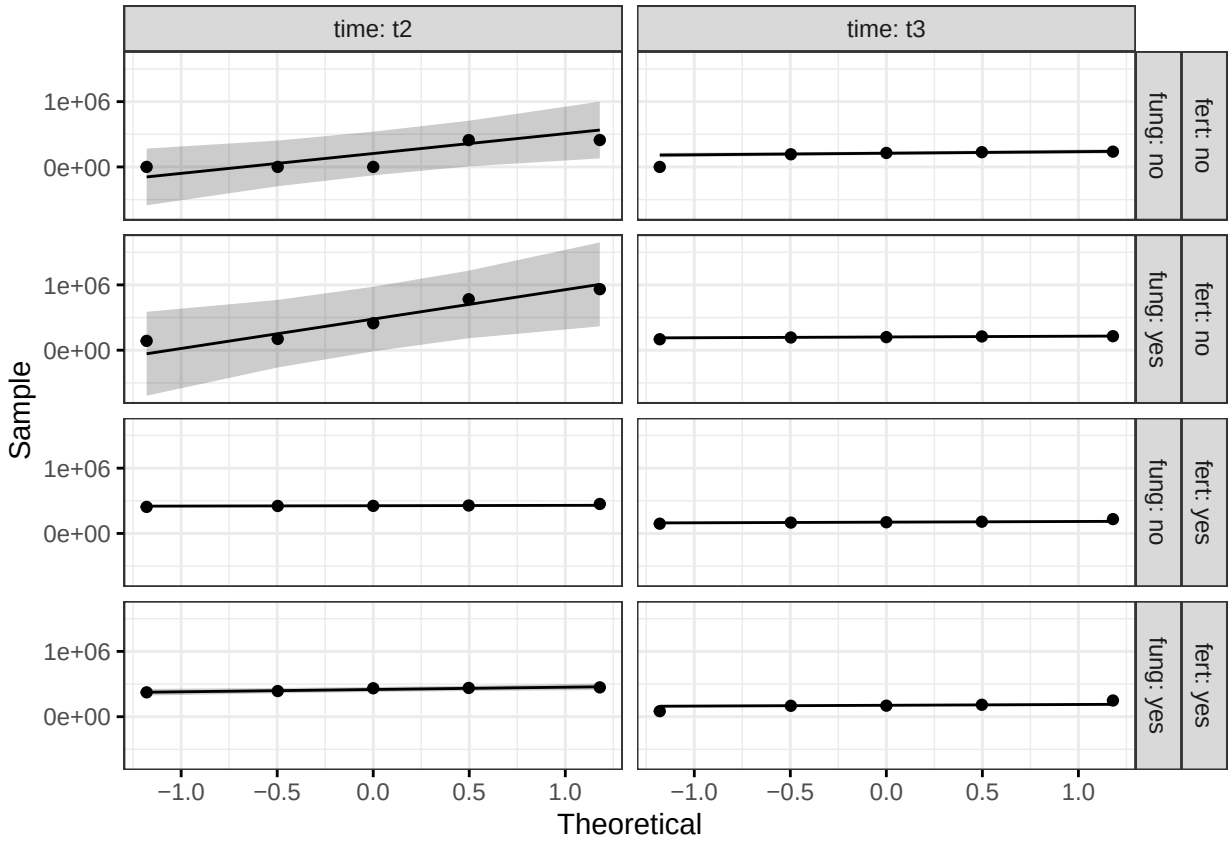
We have to check normality test and inspect QQplots to determine if ANOVA will be a viable option for eval here.

Normality and heteroskedasticity

```
# A tibble: 8 x 6
  fert  fung  time  variable statistic       p
  <fct> <fct> <fct> <chr>        <dbl>   <dbl>
1 no    no    t2    qty_ng       0.685 0.00666
2 no    no    t3    qty_ng       0.692 0.00787
3 no    yes   t2    qty_ng       0.890 0.355
4 no    yes   t3    qty_ng       0.898 0.398
5 yes   no    t2    qty_ng       0.933 0.616
6 yes   no    t3    qty_ng       0.914 0.495
7 yes   yes   t2    qty_ng       0.868 0.258
8 yes   yes   t3    qty_ng       0.934 0.624
```

SOIL





Two groups fail the normality test: [fert:no|fung:no|time:t2] and [fert:no|fung:no|time:t3]. Crosschecking the outlier output. In group [fert:no|fung:no|time:t3] we can exclude 382 which is an extreme outlier. But for the [fert:no|fung:no|time:t2] group, there is no outlier. The QQ plot give the same information as the outlier check. We have tails at the beginning and end which indicate extreme values. For QQPlots of the individual groups it looks fine.

Groups that failed the Shapiro Test and their replicates

Datatable from group [fert:no|fung:no|time:t2]

```
# A tibble: 5 x 8
```

SOIL

id	qty_ng	fert	fung	block	row	time	subject
<fct>	<dbl>	<fct>	<fct>	<fct>	<fct>	<fct>	<dbl>
1 198	411193.	no	no	b	1	t2	1
2 201	419.	no	no	c	1	t2	2
3 232	625.	no	no	d	2	t2	3
4 259	412128.	no	no	e	3	t2	4
5 286	751.	no	no	f	4	t2	5

Datatable from group [fert:no|fung:no|time:t3]

A tibble: 5 x 8

id	qty_ng	fert	fung	block	row	time	subject
<fct>	<dbl>	<fct>	<fct>	<fct>	<fct>	<fct>	<dbl>
1 294	226141.	no	no	b	1	t3	1
2 297	193851.	no	no	c	1	t3	2
3 328	234292.	no	no	d	2	t3	3
4 355	214247.	no	no	e	3	t3	4
5 382	99.0	no	no	f	4	t3	5

Possible reasons why the gene copy numbers differ so much For [fert:no|fung:no|time:t2] 198 and 259 have very high 16S gene copy numbers compared to the rest of the group. I have checked the raw values from the qPCR machine and they are correct. There are a few possibilities why the values are so high:

- The wells in the qPCR plate of 198 are close the standard, might be cross contaminated with standard (although the other replicates had the same proximity to the standard wells)
- The DNA extract was cross-contaminated with another DNA extract with higher gene copy numbers
- High heterogeneity of bacteria numbers in soil
- Sampled from a larger rhizodeposition where bacteria thrive

For '[fert:no|fung:no|time:t3]' it is possibly enough to exclude 382 from the outlier list. For the repeated test 198, 259 and 382 are excluded.

```
# A tibble: 8 x 6
  fert  fung  time  variable statistic     p
  <fct> <fct> <fct> <chr>        <dbl> <dbl>
1 no    no    t2    qty_ng       0.981 0.738
2 no    no    t3    qty_ng       0.957 0.762
3 no    yes   t2    qty_ng       0.890 0.355
4 no    yes   t3    qty_ng       0.898 0.398
5 yes   no    t2    qty_ng       0.933 0.616
6 yes   no    t3    qty_ng       0.914 0.495
7 yes   yes   t2    qty_ng       0.868 0.258
8 yes   yes   t3    qty_ng       0.934 0.624
```

Choosing a stastical model and performing ANOVA

An analysis of the dataset structure is needed to find the right statistical model. The data was generated in a randomized complete block design (RCBD).

SOIL

Model parameters

Fixed effects

- Fertilizer (fert) (Binary variable (yes/no))
- Fungicide and growth (fung) (Binary variable (yes/no))

Random effects

- Time
- Block
- (Row)
- (Could possibly include rainfall 3-7 days before sampling)

Mixed effect model

The model needs to account for the above listed fixed and random effects. The regular lm() function of R stats package will fit all variables as fixed effects if they are integrated into the forumlae. Therefore we need the package nlme which can account for random effects. Because we are using only two timepoints I will stick to a linear model. Generally, I'd consider a non-linear model if all timepoints would be in the analysis. We are investigating gene copy numbers which are directly correlated and have a causal relation ship with number of bacteria. Bacteria growth is better estimated with a logistic regression.

I'll do a stepwise modelling approach without the mathematic forumlae (will be done in the thesis tho).

All main effects + Interaction (Full model)

```
Linear mixed-effects model fit by REML
  Data: genecopy.work.filtered
      AIC      BIC    logLik
  836.957 848.1666 -410.4785

Random effects:
 Formula: ~1 | block
        (Intercept)
StdDev:    30.62974

 Formula: ~1 | time %in% block
        (Intercept)
StdDev:    37576.01

 Formula: ~1 | row %in% time %in% block
        (Intercept) Residual
StdDev:    121044.1 147502.9

Fixed effects:  qty_ng ~ fert + fung + fert * fung
                    Value Std.Error DF   t-value p-value
(Intercept)      174709.5  70450.41 14  2.479894  0.0265
fertyes          116325.3  86854.93 14  1.339305  0.2018
fungyes          153301.8  83429.38 14  1.837504  0.0874
fertyes:fungyes -125423.3 110614.30 14 -1.133880  0.2759
```

SOIL

```
 Correlation:
                (Intr) fertys fungys
fertyes         -0.705
fungyes         -0.714  0.641
fertyes:fungyes  0.527 -0.746 -0.734

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-1.3618387 -0.5626442 -0.1996662  0.4797629  2.3432915

Number of Observations: 34
Number of Groups:
                   block          time %in% block row %in% time %in% block
                       5                        10                        17
```

Reduced model (both main effects)

```
Linear mixed-effects model fit by REML
  Data: genecopy.work.filtered
     AIC      BIC   logLik
  860.97 869.5739 -424.485

Random effects:
 Formula: ~1 | block
        (Intercept)
StdDev:    21.16666
```

```
 Formula: ~1 | time %in% block

        (Intercept) Residual
StdDev:    65455.94   178957

Fixed effects:  qty_ng ~ fert + fung
                Value Std.Error DF   t-value p-value
(Intercept) 186043.92  60829.47 22 3.0584503  0.0058
fertyes      58646.49  62216.01 22 0.9426269  0.3561
fungyes     121378.62  62297.52 22 1.9483700  0.0642
 Correlation:
        (Intr) fertys
fertyes -0.579
fungyes -0.607  0.125

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-1.2575833 -0.6203814 -0.2079221  0.4018496  3.0342711

Number of Observations: 34
Number of Groups:
          block time %in% block
              5              10
```

Reduced model 2 (only fung)

SOIL

```
Linear mixed-effects model fit by REML
  Data: genecopy.work.filtered
       AIC      BIC    logLik
  883.7451 891.0738 -436.8726

Random effects:
 Formula: ~1 | block
        (Intercept)
StdDev:    23.87227

 Formula: ~1 | time %in% block
        (Intercept) Residual
StdDev:    74228.73 176172.7

Fixed effects:  qty_ng ~ fung
               Value Std.Error DF  t-value p-value
(Intercept) 220215.5  50243.92 23 4.382929  0.0002
fungyes     112582.4  60916.43 23 1.848145  0.0775
 Correlation:
        (Intr)
fungyes -0.644

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.51351777 -0.63663387 -0.05071989  0.47593626  2.81534956
```

```
Number of Observations: 34

Number of Groups:

          block time %in% block

              5              10
```

Fungicide + Growth thingy is borderline significant. May be worth to do post-hoc tests

Chapter 7

Climate data

Close to experimental Site (53°21'58.5"N|13°48'13.3"E). Data extracted from wetterkontor.com from weather station in Grünow (53°19'01.7"N|13°56'55.0"E)

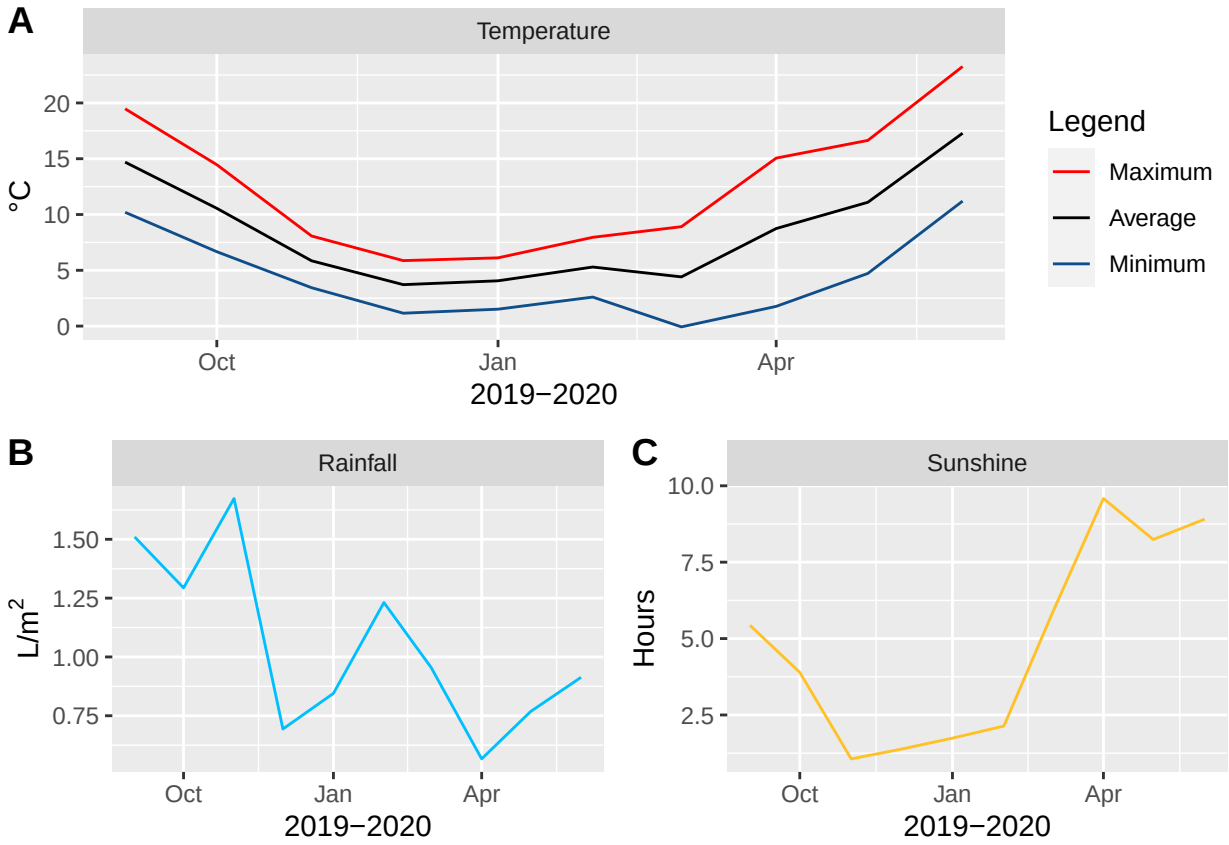
```
#Read data as tibble (tidyverse)
  climate <-
    read_delim("C:/Users/jjohn/OneDrive/MScthesis/data/clim.csv",";",) %>%
    filter (date != 0) #deletes empty days (artifact from data mining)
```

```
-- Column specification --------------------------------------------------------
cols(
  date = col_character(),
  value = col_double(),
  id = col_character(),
  format = col_character()
)
```

```
  climate$date <- as.Date(climate$date,"%d.%m.%Y") #formats the date correct

#Summary statistics as we have day data, condense to month
  summary <- climate %>%
    mutate(month = format(date,"%m"), year = format (date, "%y"))%>% #new month and
    group_by(id, month, year)%>%                                      #group by new va
    summarise(
      avg = mean(value) #mean it
    )

`summarise()` regrouping output by 'id', 'month' (override with `.groups` argument)

# Plotting the weather data


  summary$id <-
    as.factor(summary$id)
  summary$time <-
    lubridate::ymd(paste0(summary$year,summary$month,"01"))#reintroducing date forma

# consistent coloring scheme
  my_color <- c ("deepskyblue1", "goldenrod1", "black", "red", "dodgerblue4")
  names(my_color) <- levels(summary$id)
  my_scale <- scale_color_manual(name = "Legend",
                                 values = my_color,
                                 breaks=c("temp_max","temp_avg", "temp_min"),
                                 labels=c("Maximum", "Average", "Minimum"))
```

```
# filtering data for separate plots

  temp <- filter(summary, id =="temp_max" |id =="temp_min" | id=="temp_avg")
  sun <- filter(summary,id =="sunshine")
  rain<- filter(summary,id=="rainfall")



# ggplot area
  #temperature
  temp$title <- "Temperature"
  a <- ggplot(temp, aes(time,avg,color=id)) +
    geom_line() +
    ylab("°C")+
    xlab("2019-2020")+
    facet_grid(~title)+
    NULL
  plot_temp <- a + my_scale

  #rainfall, sunshine
  rain$title <- "Rainfall"
  b <- ggplot(rain, aes(time,avg, color=id)) +
    geom_line() +
    ylab(expression(paste("L/m"^"2")))+
    xlab("2019-2020")+
    facet_grid(~title)+
    NULL
```

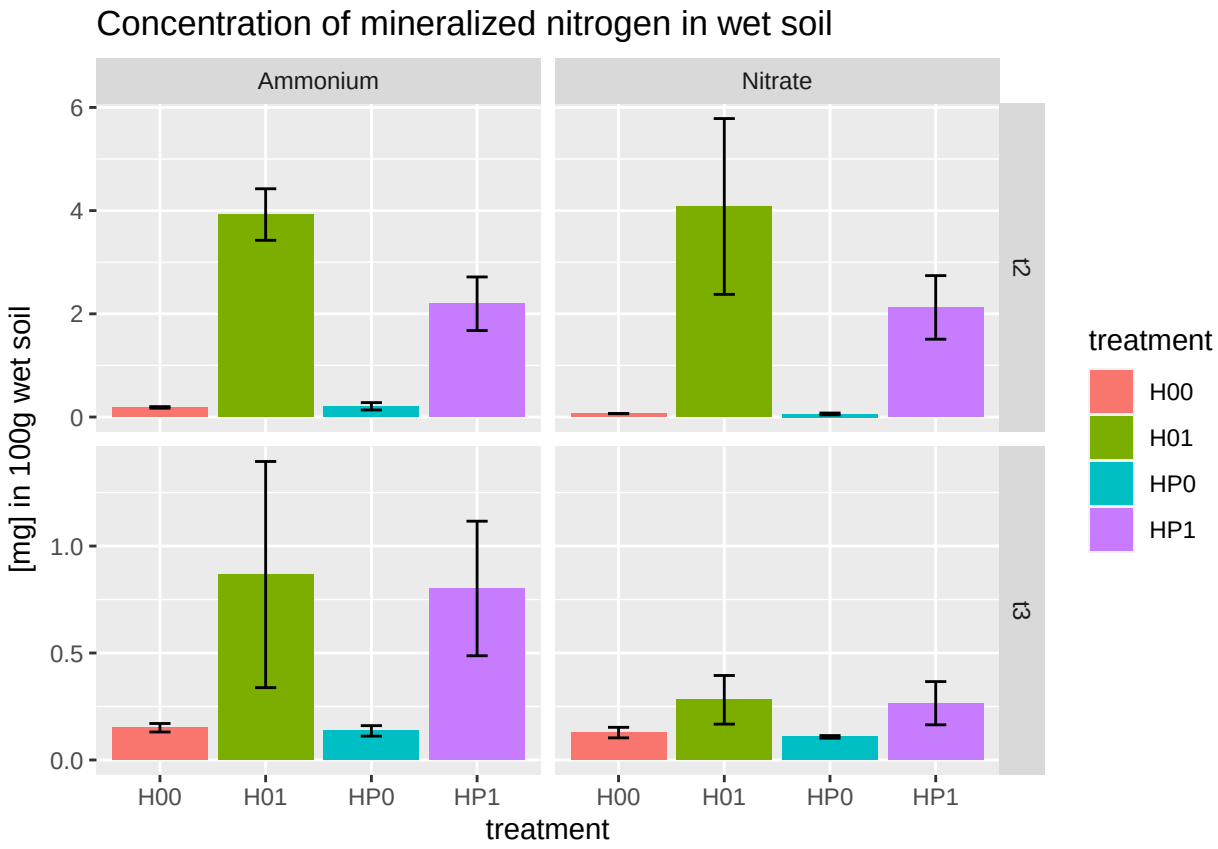
```
  plot_rain <- b + my_scale


  sun$title <- "Sunshine"
  c <- ggplot(sun, aes(time,avg,color=id)) +
    geom_line() +
    ylab("Hours")+
    xlab("2019-2020")+
    facet_grid(~title)+
    NULL
  plot_sun <- c + my_scale



#cowplot to arrange
  #Plotting two plots together
  plot_other<- plot_grid(plot_rain + theme(legend.position="none"),
                         plot_sun + theme(legend.position="none")
                         ,labels = c('B', 'C'))
  # so they can be in one row
  prow <- plot_grid(plot_temp,
            plot_other,
            labels = c('A', ''),
            ncol = 1, nrow = 2)
  prow
```

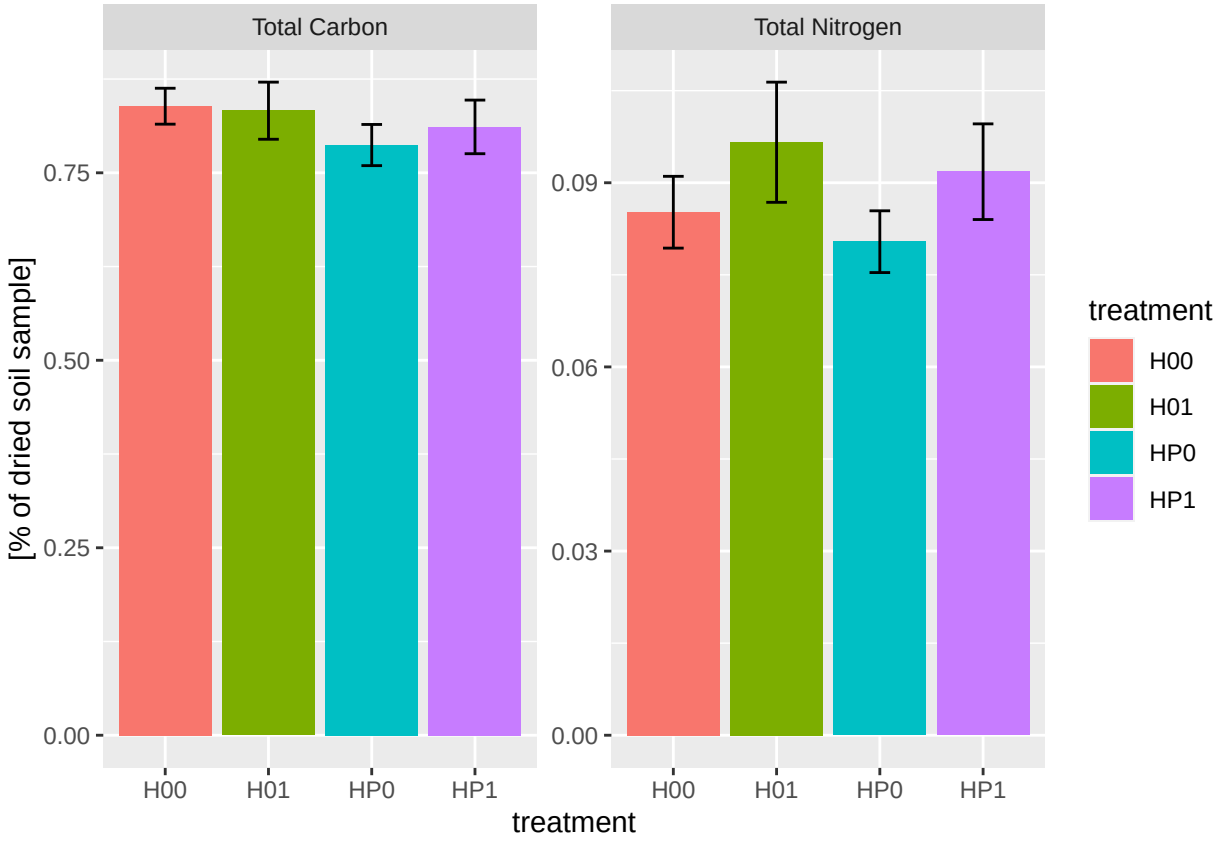


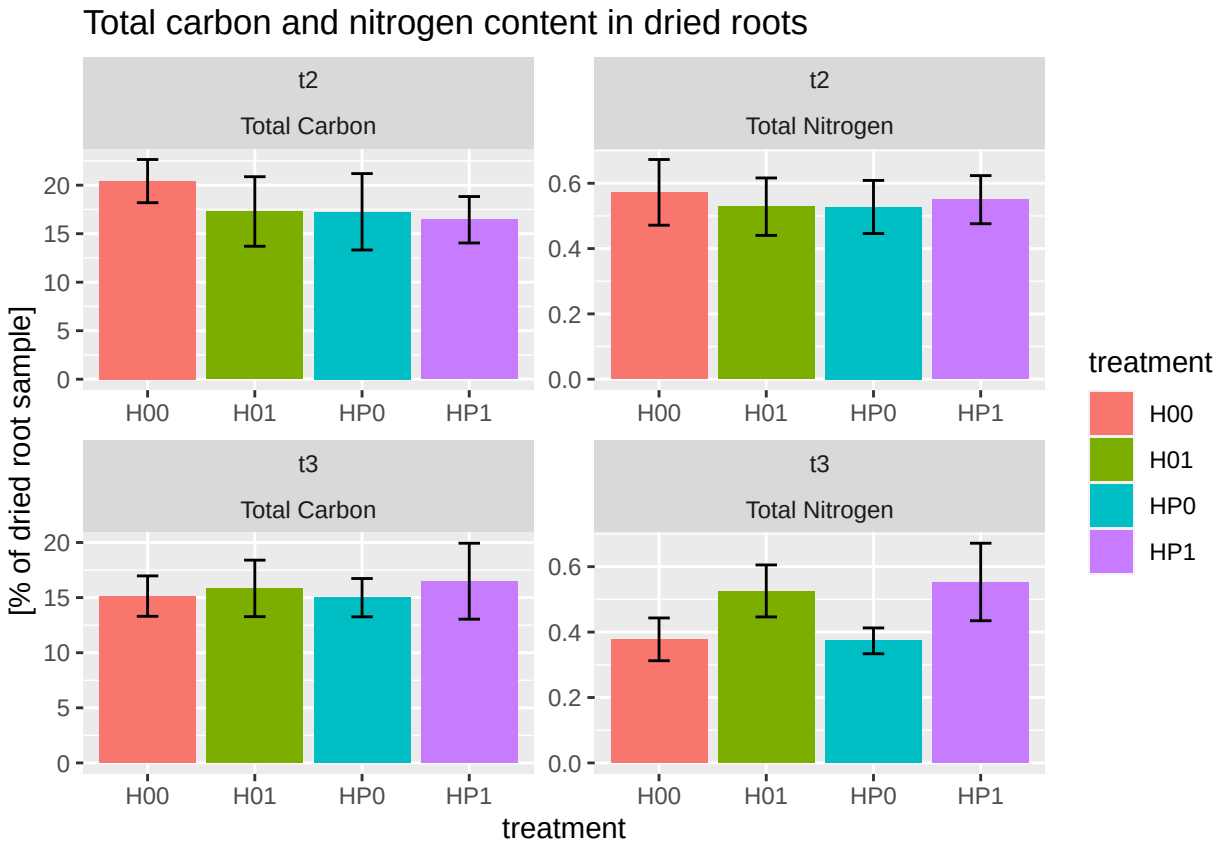

```
-- Column specification --------------------------------------------------------
cols(
  number = col_character(),
  value = col_character(),
  id = col_character(),
  unit = col_character(),
  type = col_character(),
  filename = col_character(),
  crop = col_character(),
  treatment = col_character(),
```

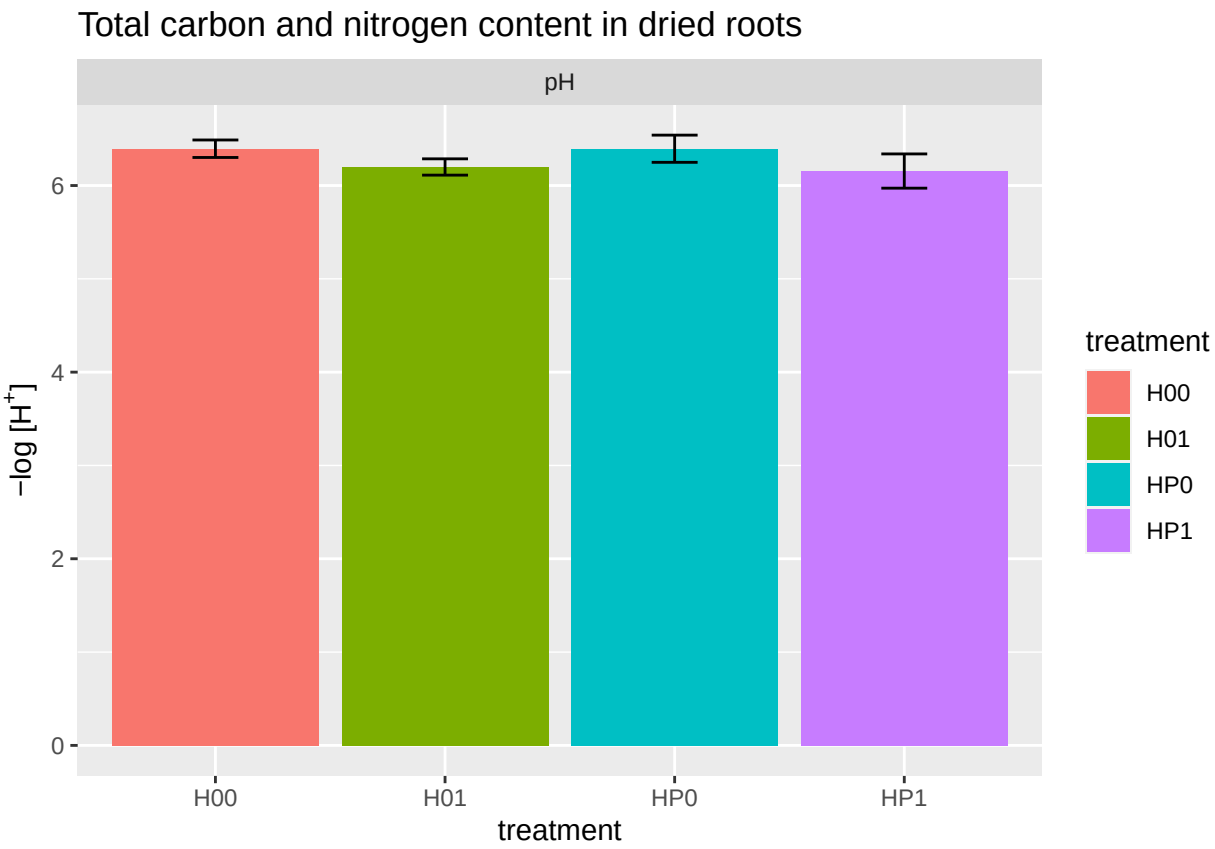
```
  timepoint = col_character()
)
```

```
`summarise()` regrouping output by 'type', 'timepoint', 'id' (override with `.groups`
```




Total Carbon
Total Nitrogen
[% of dried soil sample]
0.75
0.50
0.25
0.00
0.09
0.06
0.03
0.00
H00
H01
HP0
HP1
treatment
treatment
H00
H01
HP0
HP1


Total carbon and nitrogen content in dried roots
t2
Total Carbon
t2
Total Nitrogen
t3
Total Carbon
t3
Total Nitrogen
[% of dried root sample]
treatment
H00
H01
HP0
HP1
0
5
10
15
20
0.0
0.2
0.4
0.6



Math

TEX is the best way to typeset mathematics. Donald Knuth designed TEX when he got frustrated at how long it was taking the typesetters to finish his book, which contained a lot of mathematics. One nice feature of *R Markdown* is its ability to read LaTeX code directly.

If you are doing a thesis that will involve lots of math, you will want to read the following section which has been commented out. If you're not going to use math, skip over or delete this next commented section.

Chemistry 101: Symbols

Chemical formulas will look best if they are not italicized. Get around math mode's automatic italicizing in LaTeX by using the argument `$\mathrm{formula here}$`, with your formula inside the curly brackets. (Notice the use of the backticks here which enclose text that acts as code.)

So, $Fe_2^{2+}Cr_2O_4$ is written `$\mathrm{Fe_2^{2+}Cr_2O_4}$`.

Exponent or Superscript: O^-

Subscript: CH_4

To stack numbers or letters as in Fe_2^{2+}, the subscript is defined first, and then the superscript is defined.

Bullet: $CuCl \bullet 7H_2O$

Delta: Δ

Reaction Arrows: $\longrightarrow$ or $\xrightarrow{solution}$

Resonance Arrows: $\leftrightarrow$

Reversible Reaction Arrows: $\rightleftharpoons$

Typesetting reactions

You may wish to put your reaction in an equation environment, which means that LaTeX will place the reaction where it fits and will number the equations for you.

$$C_6H_{12}O_6 + 6O_2 \longrightarrow 6CO_2 + 6H_2O \tag{7.1}$$

PHYSICS

We can reference this combustion of glucose reaction via Equation (7.1).

Other examples of reactions

$$NH_4Cl_{(s)} \rightleftharpoons NH_{3(g)} + HCl_{(g)}$$

$$MeCH_2Br + Mg \xrightarrow[below]{above} MeCH_2 \bullet Mg \bullet Br$$

Physics

Many of the symbols you will need can be found on the math page http://web.reed.edu/cis/help/latex/math.html and the Comprehensive LaTeX Symbol Guide (http://mirror.utexas.edu/ctan/info/symbols/comprehensive/symbols-letter.pdf).

Biology

You will probably find the resources at http://www.lecb.ncifcrf.gov/~toms/latex.html helpful, particularly the links to bsts for various journals. You may also be interested in TeXShade for nucleotide typesetting (http://homepages.uni-tuebingen.de/beitz/txe.html). Be sure to read the proceeding chapter on graphics and tables.

Chapter 8

Tables, Graphics, References, and Labels

Tables

By far the easiest way to present tables in your thesis is to store the contents of the table in a CSV or Excel file, then read that file in to your R Markdown document as a data frame. Then you can style the table with the `kable` function, or functions in the kableExtra pacakge.

In addition to the tables that can be automatically generated from a data frame in **R** that you saw in R Markdown Basics using the `kable` function, you can also create tables using *pandoc*. (More information is available at `http://pandoc.org/README.html#tables`.) This might be useful if you don't have values specifically stored in **R**, but you'd like to display them in table form. Below is an example. Pay careful attention to the alignment in the table and hyphens to create the rows and columns. Generally I don't recommend this approach of typing the table directly into your R Markdown document.

Table 8.1: Correlation of Inheritance Factors for Parents and Child

Factors	Correlation between Parents & Child	Inherited
Education	-0.49	Yes
Socio-Economic Status	0.28	Slight
Income	0.08	No
Family Size	0.18	Slight
Occupational Prestige	0.21	Slight

We can also create a link to the table by doing the following: Table 8.1. If you go back to Loading and exploring data and look at the `kable` table, we can create a reference to this max delays table too: Table 2.1. The addition of the `(\#tab:inher)` option to the end of the table caption allows us to then make a reference to Table `\@ref(tab:label)`. Note that this reference could appear anywhere throughout the document after the table has appeared.

Figure 8.1: UW logo

Figures

If your thesis has a lot of figures, *R Markdown* might behave better for you than that other word processor. One perk is that it will automatically number the figures accordingly in each chapter. You'll also be able to create a label for each figure, add a caption, and then reference the figure in a way similar to what we saw with tables earlier. If you label your figures, you can move the figures around and *R Markdown* will automatically adjust the numbering for you. No need for you to remember! So that you don't have to get too far into LaTeX to do this, a couple **R** functions have been created for you to assist. You'll see their use below.

In the **R** chunk below, we will load in a picture stored as `uw.png` in our main directory. We then give it the caption of "UW logo", the label of "uwlogo", and specify that this is a figure. Make note of the different **R** chunk options that are given in the R Markdown file (not shown in the knitted document).

```
include_graphics(path = "figure/uw.png")
```

Here is a reference to the UW logo: Figure 8.1. Note the use of the `fig:` code here. By naming the **R** chunk that contains the figure, we can then reference that figure later as done in the first sentence here. We can also specify the caption for the figure via the R chunk option `fig.cap`.

FIGURES

Below we will investigate how to save the output of an **R** plot and label it in a way similar to that done above. Recall the `flights` dataset from Chapter 2. (Note that we've shown a different way to reference a section or chapter here.) We will next explore a bar graph with the mean flight departure delays by airline from Portland for 2014. Note also the use of the `scale` parameter which is discussed on the next page.

```
flights %>% group_by(carrier) %>%
  summarize(mean_dep_delay = mean(dep_delay)) %>%
  ggplot(aes(x = carrier, y = mean_dep_delay)) +
  geom_bar(position = "identity", stat = "identity", fill = "red")
```

```
`summarise()` ungrouping output (override with `.groups` argument)
```

Here is a reference to this image: Figure 8.2.

A table linking these carrier codes to airline names is available at https://github.com/ismayc/pnwflights14/blob/master/data/airlines.csv.

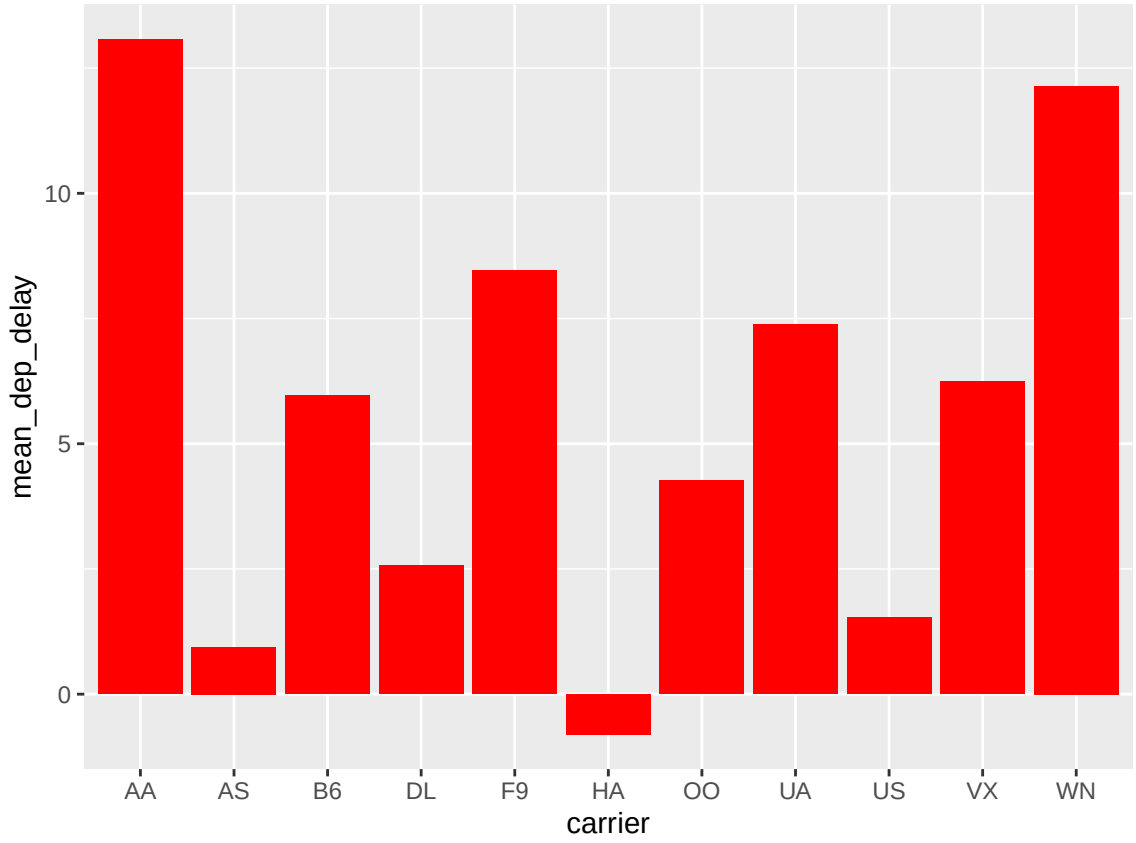


Figure 8.2: Mean Delays by Airline

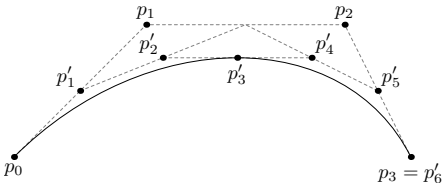


Figure 8.3: Subdiv. graph

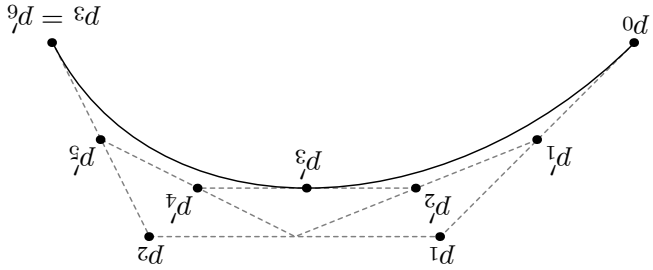


Figure 8.4: A Larger Figure, Flipped Upside Down

Next, we will explore the use of the `out.extra` chunk option, which can be used to shrink or expand an image loaded from a file by specifying `"scale= "`. Here we use the mathematical graph stored in the "subdivision.pdf" file. Here is a reference to this image: Figure 8.3. Note that `echo=FALSE` is specified so that the **R** code is hidden in the document.

More Figure Stuff

Lastly, we will explore how to rotate and enlarge figures using the `out.extra` chunk option. (Currently this only works in the PDF version of the book.) As another example, here is a reference: Figure 8.4.

Footnotes and Endnotes

You might want to footnote something.[1] The footnote will be in a smaller font and placed appropriately. Endnotes work in much the same way.

[1] footnote text

Bibliographies

Of course you will need to cite things, and you will probably accumulate an armful of sources. There are a variety of tools available for creating a bibliography database (stored with the .bib extension). In addition to BibTeX suggested below, you may want to consider using the free and easy-to-use tool called Zotero. Some Zotero documentation is at http://libguides.reed.edu/citation/zotero. In addition, a tutorial is available from Middlebury College at http://sites.middlebury.edu/zoteromiddlebury/.

R Markdown uses *pandoc* (http://pandoc.org/) to build its bibliographies. One nice caveat of this is that you won't have to do a second compile to load in references as standard LaTeX requires. To cite references in your thesis (after creating your bibliography database), place the reference name inside square brackets and precede it by the "at" symbol. For example, here's a reference to a book about worrying: [1]. This `Molina1994` entry appears in a file called `thesis.bib` in the `bib` folder. This bibliography database file was created by a program called BibTeX. You can call this file something else if you like (look at the YAML header in the main .Rmd file) and, by default, is to placed in the `bib` folder.

For more information about BibTeX and bibliographies, see (http://web.reed.edu/cis/help/latex/index.html)[2]. There are three pages on this topic: *bibtex* (which talks about using BibTeX, at http://web.reed.edu/cis/help/latex/bibtex.html), *bibtexstyles* (about how to find and use the bibliography style that best suits your needs, at http://web.reed.edu/cis/help/latex/bibtexstyles.html) and *bibman* (which covers how to make and maintain a bibliography by hand, without BibTeX, at http://web.reed.edu/cis/help/latex/bibman.html). The last page will not be useful un-

[2] 2

ANYTHING ELSE?

less you have only a few sources.

If you look at the YAML header at the top of the main .Rmd file you can see that we can specify the style of the bibliography by referencing the appropriate csl file. You can download a variety of different style files at https://www.zotero.org/styles. Make sure to download the file into the csl folder.

Tips for Bibliographies

- Like with thesis formatting, the sooner you start compiling your bibliography for something as large as thesis, the better.
- The cite key (a citation's label) needs to be unique from the other entries.
- When you have more than one author or editor, you need to separate each author's name by the word "and" e.g. `Author = {Noble, Sam and Youngberg, Jessica},`.
- Bibliographies made using BibTeX (whether manually or using a manager) accept LaTeX markup, so you can italicize and add symbols as necessary.
- To force capitalization in an article title or where all lowercase is generally used, bracket the capital letter in curly braces.

Anything else?

If you'd like to see examples of other things in this template, please contact us (email bmarwick@uw.edu) with your suggestions. We love to see people using *R Markdown* for their theses, and are happy to help.

Conclusion

If we don't want Conclusion to have a chapter number next to it, we can add the `{-}` attribute.

More info

And here's some other random info: the first paragraph after a chapter title or section head *shouldn't be* indented, because indents are to tell the reader that you're starting a new paragraph. Since that's obvious after a chapter or section title, proper typesetting doesn't add an indent there.

Appendix A

The First Appendix

This first appendix includes all of the R chunks of code that were hidden throughout the document (using the `include = FALSE` chunk tag) to help with readibility and/or setup.

In the main Rmd file

```
# This chunk ensures that the gauchodown package is
# installed and loaded. This gauchodown package includes
# the template files for the thesis.
if(!require(devtools))
  install.packages("devtools", repos = "http://cran.rstudio.com")
if(!require(gauchodown))
  devtools::install_github("danovando/gauchodown")
library(gauchodown)
```

In Chapter 8:

```
# This chunk ensures that the huskydown package is
# installed and loaded. This huskydown package includes
```

```
# the template files for the thesis and also two functions
# used for labeling and referencing
if(!require(devtools))
  install.packages("devtools", repos = "http://cran.rstudio.com")
if(!require(dplyr))
    install.packages("dplyr", repos = "http://cran.rstudio.com")
if(!require(ggplot2))
    install.packages("ggplot2", repos = "http://cran.rstudio.com")
if(!require(ggplot2))
    install.packages("bookdown", repos = "http://cran.rstudio.com")
if(!require(gauchodown)){
  library(devtools)
  devtools::install_github("benmarwick/gauchodown")
  }
library(gauchodown)
flights <- read.csv("data/flights.csv")
```

Appendix B

The Second Appendix, for Fun

Colophon

This document is set in EB Garamond, Source Code Pro and Lato. The body text is set at 11pt with *lmr*.

It was written in R Markdown and LaTeX, and rendered into PDF using gauchodown and bookdown.

This document was typeset using the XeTeX typesetting system, and the University of Washington Thesis class class created by Jim Fox. Under the hood, the University of Washington Thesis LaTeX template is used to ensure that documents conform precisely to submission standards. Other elements of the document formatting source code have been taken from the Latex, Knitr, and RMarkdown templates for UC Berkeley's graduate thesis, and Dissertate: a LaTeX dissertation template to support the production and typesetting of a PhD dissertation at Harvard, Princeton, and NYU

The source files for this thesis, along with all the data files, have been organised into an R package, xxx, which is available at `https://github.com/xxx/xxx`. A hard copy of the thesis can be found in the University of Washington library.

This version of the thesis was generated on 2021-01-07 12:00:53. The repository is currently at this commit:

The computational environment that was used to generate this version is as follows:

```
- Session info -------------------------------------------------------------
 setting  value
 version  R version 4.0.3 (2020-10-10)
 os       Windows 10 x64
 system   x86_64, mingw32
 ui       RTerm
 language (EN)
 collate  English_United States.1252
 ctype    English_United States.1252
 tz       Europe/Berlin
 date     2021-01-07

- Packages -----------------------------------------------------------------
 package     * version date       lib source
 abind         1.4-5   2016-07-21 [1] CRAN (R 4.0.3)
 assertthat    0.2.1   2019-03-21 [1] CRAN (R 4.0.3)
 backports     1.2.0   2020-11-02 [1] CRAN (R 4.0.3)
 bookdown      0.21    2020-10-13 [1] CRAN (R 4.0.3)
 boot          1.3-25  2020-04-26 [2] CRAN (R 4.0.3)
 broom         0.7.3   2020-12-16 [1] CRAN (R 4.0.3)
 callr         3.5.1   2020-10-13 [1] CRAN (R 4.0.3)
 car           3.0-10  2020-09-29 [1] CRAN (R 4.0.3)
 carData       3.0-4   2020-05-22 [1] CRAN (R 4.0.3)
 cellranger    1.1.0   2016-07-27 [1] CRAN (R 4.0.3)
 cli           2.2.0   2020-11-20 [1] CRAN (R 4.0.3)
 colorspace    2.0-0   2020-11-11 [1] CRAN (R 4.0.3)
```

```
cowplot     * 1.1.1   2020-12-30 [1] CRAN (R 4.0.3)
crayon        1.3.4   2017-09-16 [1] CRAN (R 4.0.3)
curl          4.3     2019-12-02 [1] CRAN (R 4.0.3)
data.table    1.13.4  2020-12-08 [1] CRAN (R 4.0.3)
DBI           1.1.0   2019-12-15 [1] CRAN (R 4.0.3)
dbplyr        2.0.0   2020-11-03 [1] CRAN (R 4.0.3)
desc          1.2.0   2018-05-01 [1] CRAN (R 4.0.3)
devtools    * 2.3.2   2020-09-18 [1] CRAN (R 4.0.3)
digest        0.6.27  2020-10-24 [1] CRAN (R 4.0.3)
dplyr       * 1.0.2   2020-08-18 [1] CRAN (R 4.0.3)
ellipsis      0.3.1   2020-05-15 [1] CRAN (R 4.0.3)
evaluate      0.14    2019-05-28 [1] CRAN (R 4.0.3)
fansi         0.4.1   2020-01-08 [1] CRAN (R 4.0.3)
farver        2.0.3   2020-01-16 [1] CRAN (R 4.0.3)
forcats     * 0.5.0   2020-03-01 [1] CRAN (R 4.0.3)
foreign       0.8-81  2020-12-22 [2] CRAN (R 4.0.3)
fs            1.5.0   2020-07-31 [1] CRAN (R 4.0.3)
gauchodown  * 1.0     2021-01-07 [1] Github (danovando/gauchodown@d9a19b8)
generics      0.1.0   2020-10-31 [1] CRAN (R 4.0.3)
ggplot2     * 3.3.3   2020-12-30 [1] CRAN (R 4.0.3)
ggpubr      * 0.4.0   2020-06-27 [1] CRAN (R 4.0.3)
ggsci         2.9     2018-05-14 [1] CRAN (R 4.0.3)
ggsignif      0.6.0   2019-08-08 [1] CRAN (R 4.0.3)
git2r         0.27.1  2020-05-03 [1] CRAN (R 4.0.3)
glue          1.4.2   2020-08-27 [1] CRAN (R 4.0.3)
gtable        0.3.0   2019-03-25 [1] CRAN (R 4.0.3)
```

```
haven         2.3.1   2020-06-01 [1] CRAN (R 4.0.3)
highr         0.8     2019-03-20 [1] CRAN (R 4.0.3)
hms           0.5.3   2020-01-08 [1] CRAN (R 4.0.3)
htmltools     0.5.0   2020-06-16 [1] CRAN (R 4.0.3)
httr          1.4.2   2020-07-20 [1] CRAN (R 4.0.3)
jsonlite      1.7.2   2020-12-09 [1] CRAN (R 4.0.3)
knitr       * 1.30    2020-09-22 [1] CRAN (R 4.0.3)
labeling      0.4.2   2020-10-20 [1] CRAN (R 4.0.3)
lattice       0.20-41 2020-04-02 [2] CRAN (R 4.0.3)
lifecycle     0.2.0   2020-03-06 [1] CRAN (R 4.0.3)
lme4        * 1.1-26  2020-12-01 [1] CRAN (R 4.0.3)
lubridate     1.7.9.2 2020-11-13 [1] CRAN (R 4.0.3)
magrittr      2.0.1   2020-11-17 [1] CRAN (R 4.0.3)
MASS          7.3-53  2020-09-09 [2] CRAN (R 4.0.3)
Matrix      * 1.2-18  2019-11-27 [2] CRAN (R 4.0.3)
memoise       1.1.0   2017-04-21 [1] CRAN (R 4.0.3)
minqa         1.2.4   2014-10-09 [1] CRAN (R 4.0.3)
modelr        0.1.8   2020-05-19 [1] CRAN (R 4.0.3)
munsell       0.5.0   2018-06-12 [1] CRAN (R 4.0.3)
nlme        * 3.1-151 2020-12-10 [2] CRAN (R 4.0.3)
nloptr        1.2.2.2 2020-07-02 [1] CRAN (R 4.0.3)
openxlsx      4.2.3   2020-10-27 [1] CRAN (R 4.0.3)
pillar        1.4.7   2020-11-20 [1] CRAN (R 4.0.3)
pkgbuild      1.2.0   2020-12-15 [1] CRAN (R 4.0.3)
pkgconfig     2.0.3   2019-09-22 [1] CRAN (R 4.0.3)
pkgload       1.1.0   2020-05-29 [1] CRAN (R 4.0.3)
```

```
plyr          1.8.6   2020-03-03 [1] CRAN (R 4.0.3)
prettyunits   1.1.1   2020-01-24 [1] CRAN (R 4.0.3)
processx      3.4.5   2020-11-30 [1] CRAN (R 4.0.3)
ps            1.5.0   2020-12-05 [1] CRAN (R 4.0.3)
purrr       * 0.3.4   2020-04-17 [1] CRAN (R 4.0.3)
R6            2.5.0   2020-10-28 [1] CRAN (R 4.0.3)
Rcpp          1.0.5   2020-07-06 [1] CRAN (R 4.0.3)
readr       * 1.4.0   2020-10-05 [1] CRAN (R 4.0.3)
readxl        1.3.1   2019-03-13 [1] CRAN (R 4.0.3)
remotes       2.2.0   2020-07-21 [1] CRAN (R 4.0.3)
reprex        0.3.0   2019-05-16 [1] CRAN (R 4.0.3)
reshape2    * 1.4.4   2020-04-09 [1] CRAN (R 4.0.3)
rio           0.5.16  2018-11-26 [1] CRAN (R 4.0.3)
rlang         0.4.10  2020-12-30 [1] CRAN (R 4.0.3)
rmarkdown     2.6     2020-12-14 [1] CRAN (R 4.0.3)
rprojroot     2.0.2   2020-11-15 [1] CRAN (R 4.0.3)
rstatix     * 0.6.0   2020-06-18 [1] CRAN (R 4.0.3)
rstudioapi    0.13    2020-11-12 [1] CRAN (R 4.0.3)
rvest         0.3.6   2020-07-25 [1] CRAN (R 4.0.3)
scales        1.1.1   2020-05-11 [1] CRAN (R 4.0.3)
sessioninfo   1.1.1   2018-11-05 [1] CRAN (R 4.0.3)
statmod       1.4.35  2020-10-19 [1] CRAN (R 4.0.3)
stringi       1.5.3   2020-09-09 [1] CRAN (R 4.0.3)
stringr     * 1.4.0   2019-02-10 [1] CRAN (R 4.0.3)
testthat      3.0.1   2020-12-17 [1] CRAN (R 4.0.3)
tibble      * 3.0.4   2020-10-12 [1] CRAN (R 4.0.3)
```

```
 tidyr       * 1.1.2   2020-08-27 [1] CRAN (R 4.0.3)
 tidyselect    1.1.0   2020-05-11 [1] CRAN (R 4.0.3)
 tidyverse   * 1.3.0   2019-11-21 [1] CRAN (R 4.0.3)
 usethis     * 2.0.0   2020-12-10 [1] CRAN (R 4.0.3)
 utf8          1.1.4   2018-05-24 [1] CRAN (R 4.0.3)
 vctrs         0.3.6   2020-12-17 [1] CRAN (R 4.0.3)
 withr         2.3.0   2020-09-22 [1] CRAN (R 4.0.3)
 xfun          0.19    2020-10-30 [1] CRAN (R 4.0.3)
 xml2          1.3.2   2020-04-23 [1] CRAN (R 4.0.3)
 yaml          2.2.1   2020-02-01 [1] CRAN (R 4.0.3)
 zip           2.1.1   2020-08-27 [1] CRAN (R 4.0.3)

[1] D:/rlib
[2] D:/R/R-4.0.3/library
```

References

[1] S. T. Molina and T. D. Borkovec, "The Penn State worry questionnaire: Psychometric properties and associated characteristics," in *Worrying: Perspectives on theory, assessment and treatment*, G. C. L. Davey and F. Tallis, Eds. New York: Wiley, 1994, pp. 265–283.

[2] Reed College, "LaTeX your document." 2007 [Online]. Available: http://web.reed.edu/cis/help/LaTeX/index.html

[3] E. Angel, *Interactive computer graphics : A top-down approach with opengl*. Boston, MA: Addison Wesley Longman, 2000.

[4] E. Angel, *Batch-file computer graphics : A bottom-up approach with quicktime*. Boston, MA: Wesley Addison Longman, 2001.

[5] E. Angel, *Test second book by angel*. Boston, MA: Wesley Addison Longman, 2001.