Statistics Tutorials & Templates

Contents

Preface

This website contains tutorials and templates for statistical analysis in R markdown. It is primarily meant as reference material for the statistics course taught at the Institute of Biology Leiden, Leiden University.

This is not a general reference for how to do statistical analysis. That cannot be done concisely for a number of reasons:

- The right analysis depends on the goal of the study. Is the study exploratory? Is there a predefined hypothesis to be tested? Is the main goal prediction? Is the main goal (causal) inference? Has the study been properly designed for this goal?
- Phenomena that complicate typical analysis. Usually, a substantial part of analysis involves data cleaning, getting the data in tidy format, dealing with missing data, and identifying any number of other potentially complicating factors (censoring, zero-inflation, etc.).
- Limitations of frequentist statistics.
- Limitations of *any* supposedly exhaustive list of techniques. If your problem does not easily fall into one of the categories of analysis shown here, I recommend you consult with a statistician.

Finally, please note that this page is meant to show you how to *do* things, not how statistical methods work. For that, please refer to our free online material over here.

 The tutorials website is a work in progress. Please check back later for updates.

Chapter 1

T-Test

A hypothesis test to compare two means to see if they differ significantly. Assumptions depend on the type of t-test.

Summary

1. Read the data into R ;
2. Visualize the data;
3. Choose an appropriate type of t-test ;
4. Report a conclusion;
5. Incorporating this in scientific literature.

You can download the tutorial here and the required data set here.

1.1 Read the Data into R

I recommend saving data as comma-separated values (CSV). If you prefer reading data directly from Excel, have a look here.

Details

- Save the data in a folder;
- Open RStudio and create a new R markdown file; (**File** > **New File** > **R Markdown**)
- Save your R markdown file to the same location as the data;

- Set working directory to source file location. (**Session** > **Set Working Directory** > **To Source File Location**)

```
DF <- read.csv("two-groups.csv")
```

Did that work? There are several ways to check:

```
str(DF)
summary(DF)
head(DF)
```

Explanation of the output

```
str(DF)
```

```
## 'data.frame':    60 obs. of  3 variables:
##  $ X        : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Pesticide: chr  "A" "A" "A" "A" ...
##  $ CropYield: num  101.5 98.9 105.9 100.9 101 ...
```

This command shows the structure of the data. It is a data frame with $n = 60$ observations and 2 variables. `Pesticide` is a character vector (`chr`) and `CropYield` a numeric vector (`num`).

```
##        X          Pesticide          CropYield
##  Min.   : 1.00   Length:60          Min.   : 92.90
##  1st Qu.:15.75   Class :character   1st Qu.: 98.08
##  Median :30.50   Mode  :character   Median : 99.85
##  Mean   :30.50                      Mean   : 99.96
##  3rd Qu.:45.25                      3rd Qu.:101.55
##  Max.   :60.00                      Max.   :105.90
```

```
##   X Pesticide CropYield
## 1 1         A     101.5
## 2 2         A      98.9
## 3 3         A     105.9
## 4 4         A     100.9
## 5 5         A     101.0
## 6 6         A     100.4
```

My output looks different

Then provided you did everything else correctly, the most likely reason is that your data was saved with a version of Excel where a comma is used as a decimal separator (e.g., the Dutch version). The solutions for this is simple, use `read.csv2`:

```
DF <- read.csv2("some-data-with-commas.csv")
```

If that doesn't work either, go over the details shown above, or watch the instructions in the video.

Common mistakes

- Remember to include the file extension (e.g., "two-groups**.csv**") when typing the file name.
- You cannot read Excel files (.xls, .xlsx) with this function. Instead, follow the guide here, or save your data as CSV.
- Don't open the CSV file with Excel. You don't need Excel or Google Sheets or any other program besides R and RStudio. If you have saved your data as CSV, you can close Excel.

The next step is to ensure categorical variables are read as factors. This will allow us to use generic functions like `plot` and `summary` in a more meaningful way.

Why not just use `character`?

A character vector is just strings of text, numbers and/or symbols. If you were to produce a `summary`, this happens:

```
summary(DF$Pesticide)
```

```
##    Length     Class      Mode
##        60 character character
```

It tells us this object contains 60 values, and it is stored and treated as a string of text.

The generic `plot` function doesn't even work at all:

```
plot(CropYield ~ Pesticide, DF)
```

```
Error in plot.window(...) : need finite 'xlim' values
In addition: Warning messages:
1: In xy.coords(x, y, xlabel, ylabel, log) : NAs introduced by coercion
2: In min(x) : no non-missing arguments to min; returning Inf
3: In max(x) : no non-missing arguments to max; returning -Inf
```

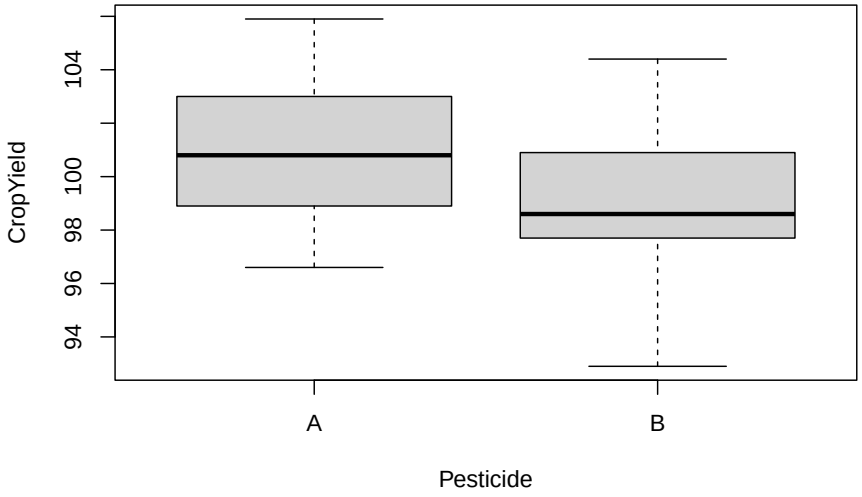
Now convert the variables to factors and see how that changes the output. *(Run the code below, then run `summary(DF$Pesticide)`, or `plot(DF$Pesticide)` for example.)*

```
DF$Pesticide <- factor(DF$Pesticide)
```

1.2 Visualize the Data

There are different ways to go about this. A basic, but still very popular choice is the boxplot:

```
boxplot(CropYield ~ Pesticide, DF)
```



What to look for

Potential Outliers

A boxplot shows either of the following:

The interquartile range (IQR) is simply the size of the box. If all observations lie within 1.5× this range from the box, then the whiskers show the extremes of the data (fig. 1.1 A). Otherwise, the observations are drawn separately 1.1 B). The IQR is usually not shown in a boxplot, but is used internally to calculate the location of the whiskers and marked observations (if any).

Marked observations are not outliers

This is a common misconception. Though it can be an *indication* of outlyingness, a boxplot alone cannot tell you whether these observations will strongly affect your analysis:

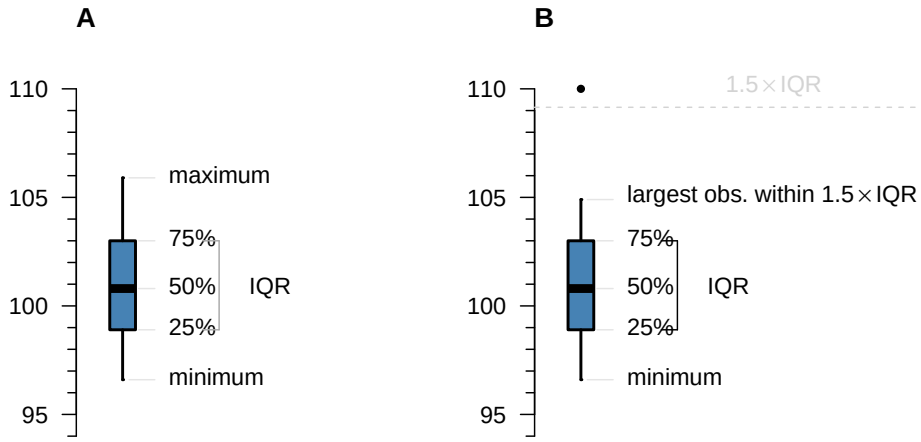


Figure 1.1: What is displayed in a boxplot in case all observations are within a certain distance from the box (A), or otherwise (B).

- If you have a large enough sample size, you *will* find more extreme observations, which are eventually drawn outside the IQR.
- Skewed values (see next section) will almost always show 'outliers' in the direction of skew, but these are unlikely to be outliers in the context of an appropriate model for skewed data.
- The 1.5× IQR rule is nothing special, it is merely convention. A boxplot is just a quick way to visualize numeric values.

Skew

Skew means the data-generating process follows a probability distribution that is not symmetric. In a boxplot, skew looks like this:

Skew is not necessarily a problem, unless it persists in the residuals of a model that assumes normally distributed errors. For an explanation of skew, see the video on probability distributions .

Differences Between Groups

A boxplot is not just a nice tool for *yourself* to inspect your data, but is also an effective tool to visually communicate the evidence for difference between groups:

From the plot you can conclude:

- Neither group appears to have outliers;
- Neither group appears skewed;

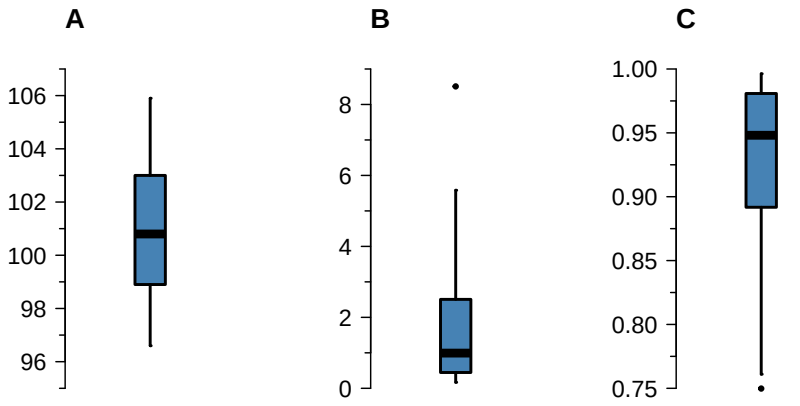


Figure 1.2: A boxplot of symmetric (A), right-skewed (B), and left-skewed values.

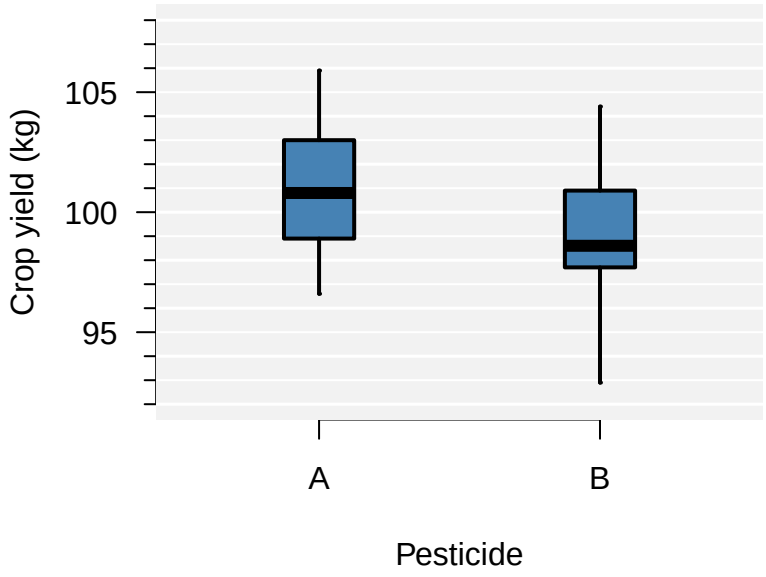


Figure 1.3: A comparison of crop yield for pesticides A and B.

- Pesticide A seems to yield more crop than B, but there is considerable overlap;
- The variance of both groups is similar, because the boxes are similarly sized.

This is just a sample, so whether this difference is *significant*, should be determined with a test.

How to improve your plot

Some basic adjustments that improve any plot:

- Use informative axis labels, with unit of measurement if appropriate;
- Add an informative caption;
- Keep the numbers on the axes horizontal where possible;
- Remove unnecessary plot elements, like the box around the figure.

The changes to the code I made below can all be found in the help pages `?boxplot` and `?par`. I am also a big fan of the `eaxis` function from the package `sfsmisc`. The caption is added as a chunk option (i.e., ```` ```{r, fig.cap = "..."} ````).

```r
# Load a package for nice axes
library("sfsmisc")

# Change the margins (bottom, left, top, right)
par(mar = c(4, 4, 0, 0) + 0.1)

# Create the coordinate system
boxplot(CropYield ~ Pesticide, DF, axes = FALSE, boxwex = 0.25, staplewex = 0,
        ylab = "Crop yield (kg)", ylim = c(92, 108), lwd = 2, lty = 1)

# Add axes
axis(1, 1:2, c("A", "B"))
eaxis(2)

# Add a lightgrey background
polygon(x = c(-1, 3, 3, -1, -1), y = c(90, 90, 110, 110, 90), col = "grey95")

# Add a simple grid
abline(h = seq(90, 110, 1), col = "white", lwd = 1)
abline(h = seq(90, 110, 2), col = "white", lwd = 1.5)

# Redraw the boxplots on top
boxplot(CropYield ~ Pesticide, DF, axes = FALSE, boxwex = 0.25, staplewex = 0,
        col = "steelblue", add = TRUE, lwd = 2, lty = 1)
```

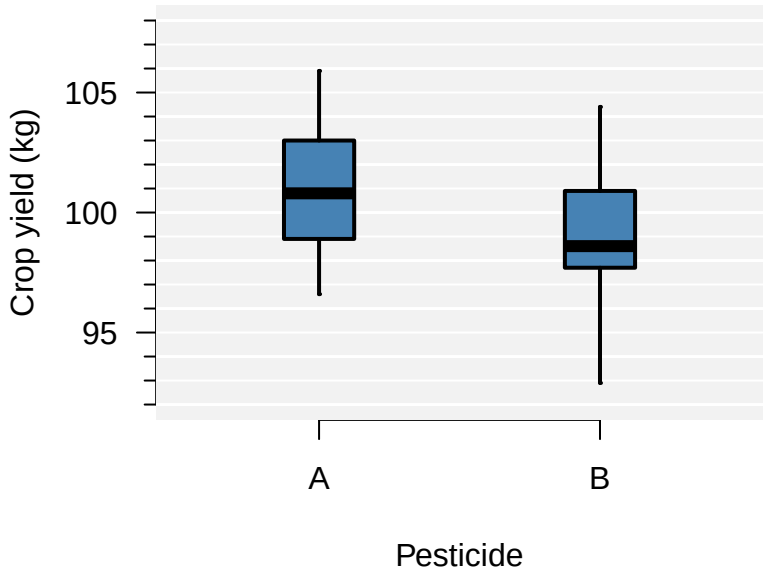


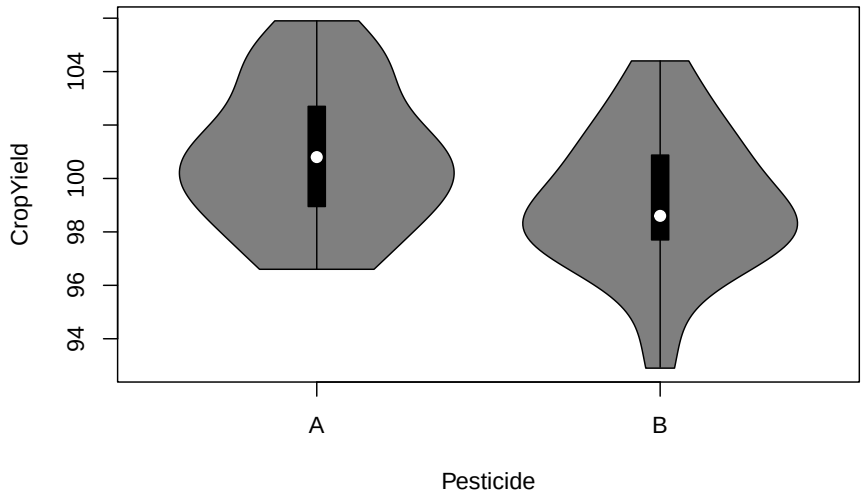
Figure 1.4: A comparison of crop yield for pesticides A and B.

```
# Restore the default margins for subsequent plots
par(mar = c(5, 4, 4, 3) + 0.1)
```

Alternative: Violin plot

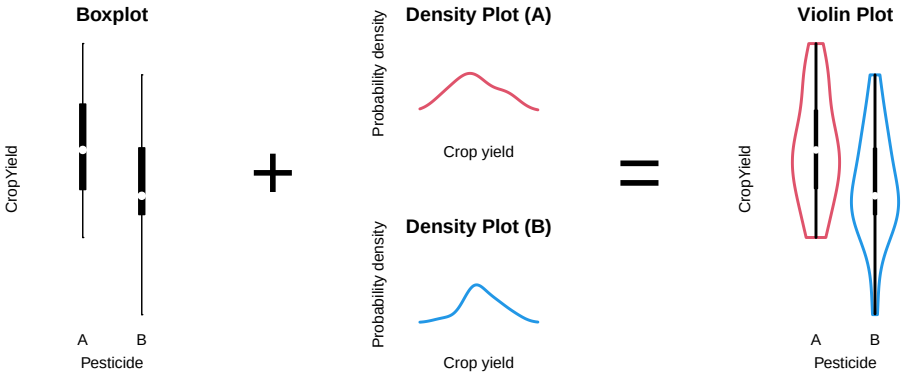
A more modern take on the boxplot is a **violin plot**. It combines a boxplot and a density plot. This type of visualization is richer in information than just a boxplot, but it is only meaningful if you have enough observations per group (e.g., >10):

```
library("vioplot") # install if missing
vioplot(CropYield ~ Pesticide, DF)
```



How to interpret a violin plot

Here is a breakdown of the components shown in a violin plot:



The density plot, if you are unfamilair with it, is like a continuous version of a histogram. It shows which values are more and less likely in the sample.

1.3 Choose an Appropriate T-Test

This part is still a work in progress. Refer to the video for now.

Table 1.1: The directions in which you can test and the corresponding null-hypothesis.

Type	Question	Syntax
two-sided	Is there a difference in group means?	't.test(..., alternative = "t
one-sided	Is the mean of A greater than that of B?	't.test(..., alternative = "g
one-sided	Is the mean of A less than that of B?	't.test(..., alternative = "l

One-sided or two-sided

You can test in two directions: Either the mean of A is greater than that of B, or the other way around. If you test for both possibilities, the test is called two-sided. Below is a short summary of the possibilities.

A one-sided test is more powerful than a two-sided test. For instance, if you want to know whether a treatment yields *lower* blood pressure than a control group, you should use a one-sided test.

If you test one-sided, you have to be able to defend this choice without seeing the data, or any plots. If you are unsure, test *two-sided*.

Equal or unequal variance

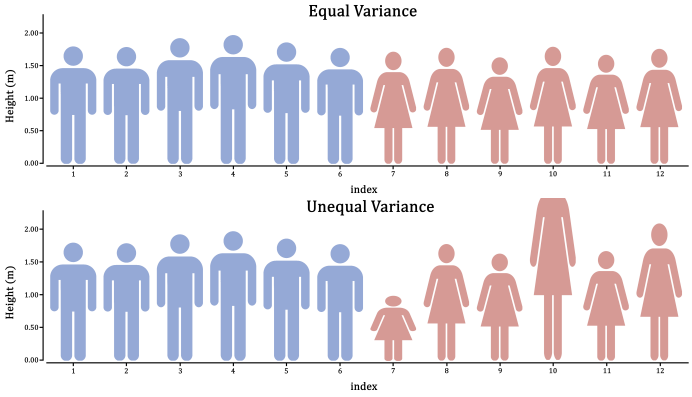


Figure 1.5: Variance is the extent to which individuals differ from their group mean. In the case of male and female human height, which do you think is more realistic?

First, think about it in the context of the research: Does it make sense for these groups to vary to the same extent; should they have similar individual differences?

If you can't think of the right answer to that question, the safer of the two assumptions is *unequal* variance. This will estimate a variance for both groups

separately, at the cost of some power.

From the boxplot we could already tell that the group variances are more or less equal in the example. In cases where it isn't obvious, you could conduct an F-test:

```
var.test(CropYield ~ Pesticide, DF)
```

```
##
##  F test to compare two variances
##
## data:  CropYield by Pesticide
## F = 1.0152, num df = 29, denom df = 29, p-value = 0.9678
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.4832202 2.1330227
## sample estimates:
## ratio of variances
##           1.015244
```

In this case, the ratio $\frac{\text{Var}(A)}{\text{Var}(B)} \approx 1$ (`1.015244`), with a p-value of `0.9678`, indicating the variances do not differ significantly.

(If these values came from populations with equal variance, you would have a 96.8% *chance of drawing a sample with at least this large a difference in variance. This chance is very large, so we do not reject the null-hypothesis.)*

One-sample t-test

It is also possible to compare a single group mean to a fixed point. For example, does the temperature in class rooms differ significantly from 20°C?

You can do this in R as follows:

```
# Measurements from 7 class rooms
Temp <- c(24, 19, 19, 23, 19, 22, 22)

# Do these measurements differ from an average of 20 degrees?
t.test(Temp, mu = 20)
```

```
##
##  One Sample t-test
##
## data:  Temp
## t = 1.4292, df = 6, p-value = 0.2029
## alternative hypothesis: true mean is not equal to 20
## 95 percent confidence interval:
```

```
##  19.18616 23.09955
## sample estimates:
## mean of x
##  21.14286
```

In this example, there is insufficient evidence to conclude a difference from 20°C (95% CI: 19.2–23.1).

Paired t-test

…

Non-parametric alternative

…

The advantage of this approach is that we use all observations (both A and B) at once to assess the assumptions. This is especially helpful when you only have a small number of observations in both groups, because diagnostics on small sample sizes is largely guesswork.

In the example shown here, the appropriate choice would be an independent, two-sample, two-sided, equal variance t-test:

```
t.test(CropYield ~ Pesticide, data = DF, var.equal = TRUE)
```

Show output

```
##
##  Two Sample t-test
##
## data:  CropYield by Pesticide
## t = 2.5604, df = 58, p-value = 0.01308
## alternative hypothesis: true difference in means between group A and group B is not
## 95 percent confidence interval:
##  0.3745639 3.0587694
## sample estimates:
## mean in group A mean in group B
##       100.82000        99.10333
```

In the output we see:

- The t-value (`t = 2.5604`);
- The degrees of freedom used to compute the p-value (`df = 58`);
- The resulting p-value (`p-value = 0.01308`); Computed as `2 * pt(2.5604, 58, lower.tail = FALSE)`
- A 95% confidence interval (0.37–3.06);
- The estimated group means ($\bar{y}_A = 100.8$, $\bar{y}_B = 99.1$).

What to do with this output is described in the next section.

1.4 Correctly Phrase the Results

If the p-value is **less** than the chosen level of significance

(This is the case in the example.)

Examples of precise language:

- Crop yield differed significantly by type of pesticide ($p = 0.0131$), with pesticide A resulting in 1.7 kg higher crop yield on average;
- Pesticide A yielded significantly higher crop yield than B ($\hat{\beta} = 1.7$ kg, $p = 0.0131$);
- Pesticide A yielded on average 1.7 kg higher crop yield than B (95% CI: 0.38–3.1);

Examples of incorrect, incomplete, or imprecise language:

- The alternative hypothesis was true // The null-hypothesis was false;
- The difference was significant ($p < 0.05$);
- Pesticide A outperforms pesticide B.

Why paste tense?

The results of a single experiment, no matter how convincing, can never prove something to be true. The results *were* observed, and in this one experiment, A *was* better than B.

Use present tense only for statements that have been demonstrated repeatedly, are generally agreed upon, or are easily observable, e.g.:

- Smoking causes cancer;
- Current climate-change is mainly caused by human activities;
- Most people use smartphones nowadays.

If the p-value is **greater** than the chosen level of significance

Examples of precise language:

- Crop yield did not differ significantly by type of pesticide ($p = ...$);
- There is insufficient evidence to conclude either pesticide worked better ($\hat{\beta} = ...$, $p = ...$);
- There was a trend towards higher crop yield with pesticide A ($\hat{\beta} = ...$), but this difference was not significant (95% CI: …).

Examples of incorrect, incomplete, or imprecise language:

- Pesticide A performed equally well as pesticide B;
- There was no difference ($p < 0.05$);
- We accept the null-hypothesis.

Why can't I say I *accepted* the null-hypothesis?

This is imprecise language because it distorts the order of null-hypothesis significance testing. Every tests *starts* with pretending the null-hypothesis is true, and then considering how rare a result this would be. You did not accept the null-hypothesis *because* of the p-value, but rather, you started by taking on the null-hypothesis to even compute that p-value.

1.5 Incorporating This in a Paper

Here you can find examples of how to justify the methods used, explain the results found and write a discussion. This is meant to show you what belongs where, and what level of detail is common in scientific literature.

Remember to use your own words—**paraphrase to avoid plagiarism**.

Methods

(In this section, you should describe the data collection and analysis to an extent that a fellow expert could reproduce your study.)

A total of $n = 60$ plots of land where corn is cultivated were selected from farms in South Holland, The Netherlands (figure …). Pesticide A or B was applied randomly in a concentration of … to 30 plots each.

All statistical analyses were conducted in R, version 4.2.1, using the RStudio interface.[R Core Team, 2022, RStudio Team, 2022] Difference in group means was assessed with an independent student's t-tests for equal variance. Conditional normality was assessed through normal quantile-quantile plots, using the `car` package.[Fox and Weisberg, 2019]

Note:

- Check your R version number with `version`;
- Justify the type of t-test used; *(For example, if you use an independent t-test, the methods section should sufficiently describe how observations were collected, such that the reader can judge whether these observations are indeed independent.)*
- If you assume unequal variance, the test is called a *Welch t-test*;
- You should cite any packages used to perform analyses, generate tables or figures that made it into the paper, e.g.;
  - Normal quantile-quantile plots were generated with the `car` package.[Fox and Weisberg, 2019]

Table 1.2: Example fictive data set of differences in gene expression differences in cases and controls. Adjusted p-values have been corrected for multiple testing using a Bonferroni correction.

Gene	Estimate	SE	t	p	p_{adjusted}
ACTB	0.107	0.354	0.302	0.382	1.000
CDH1	-0.842	0.251	-3.355	1.150×10^{-3}	5.750×10^{-3}
NF1	-0.481	0.334	-1.441	0.0805	0.4025
PTEN	0.812	1.205	0.674	0.747	1.000
TP53	3.419	0.682	5.013	1.340×10^{-5}	6.700×10^{-5}

- You should *not* cite packages used only internally, e.g.:
  - We used `readxl` to enter the data in R.[Wickham and Bryan, 2022]

Results

(In this section you should mention the results without giving any interpretation yet.)

For general advice on phrasing, see: Correctly Phrase the Results. In this section, you should include your boxplot and explain in brief what the outcome of the t-test was.

It is very common to see boxplots with significance stars, but the interpretation of these stars is not consistent and you should mention the actual, non-discretized p-value at least somewhere in the results.

When you perform multiple tests, you can summarize the results in a table, which you refer to in the results section. For example, here is how you could report the results of a qPCR experiment where 5 genes were tested for differential expression in cases and controls, and the resulting p-values were corrected for multiple testing:

Discussion

(In this section, you should not mention results, but the conclusions based on those results.)

If the test was significant **and** the difference large enough to be biologically relevant:

- For corn grown in areas of similar climate as South Holland, The Netherlands, we recommend using pesticide A;
- Further research could demonstrate whether this difference is similar in other plant species, and how it depends on other factors, such as soil composition and irrigation.

- *(Overclaiming)* Pesticide A has been proven to outperform B for corn grown in South Holland, The Netherlands.
- *(Overgeneralizing)* Pesticide A yields more crops than B.

If the test was insignificant **or** the difference too small to be biologically relevant:

- *(If the study was sufficiently powerful)* Despite a fairly large sample size, we were unable to demonstrate a difference in crop yield, suggesting both pesticides work equally well.
- *(If the study was underpowered)* We were unable to demonstrate a difference in crop yield, though a follow-up study with a larger sample size may conclude otherwise.
- *(Appeal to ignorance)* We have demonstrated there is no difference between pesticide A and B.

Why can't I conclude there is no difference?

This is an inherent limitation of the p-value. Under the null-hypothesis, any p-value is as likely as any other (uniformity). Therefore, even if the p-value is very large, this cannot be interpreted as evidence *for* the null-hypothesis.

For example, a p-value of 0.20 is just as common a result as 0.80 when the null-hypothesis is correct. The p-value can *only* be used as evidence *against* the null-hypothesis (when it is small).

If you want to perform a test to find out whether the null-hypothesis is correct, what you are looking for is called an equivalence test. For t-tests, this can be implemented easily through TOST.

Generating citations

Any packages used in analysis should be cited in the methods section. The citations used here can be easily obtained as follows (note that this requires the packages to be installed):

```
citation()        # Use this to cite R
RStudio.Version() # Use this to cite RStudio
citation("car")   # Use this to cite a package (here: car)
```

You can add these entries to a reference manager (e.g., Zotero), or keep a BibTeX file yourself, as shown here.

Supplementary

(The following is not usually found in the main text, but can be included as supplementary material.)

Including diagnostic plots can be an effective way to convince the reader you considered the assumptions of the test or model used:

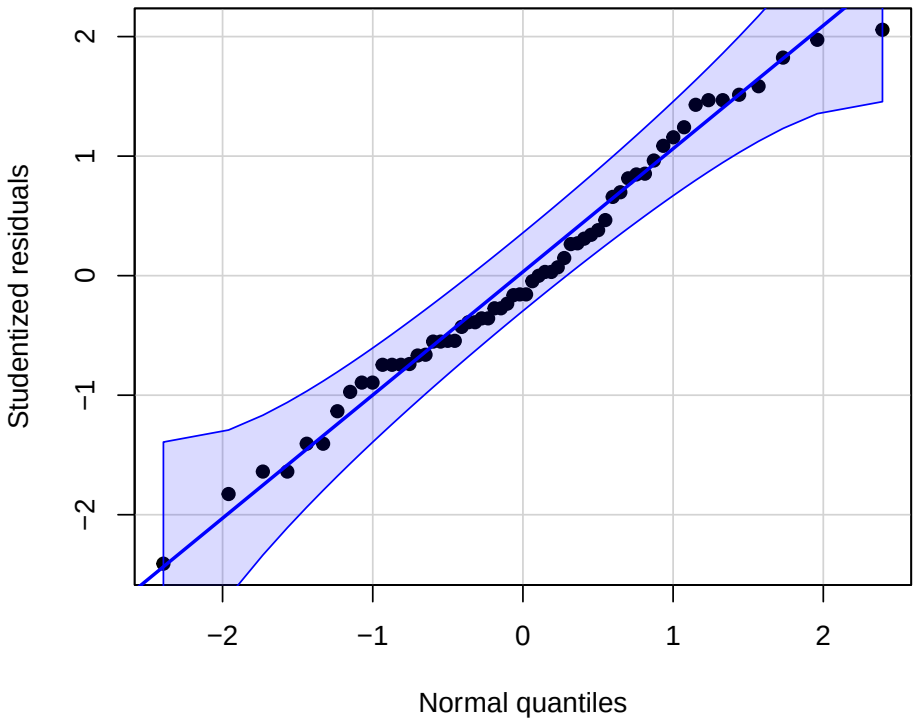


Figure 1.6: Normal quantile-quantile plot of the studentized residuals showed no notable deviation from normality.

If you include any supplementary material, you should refer to it in the main text.

Chapter 2

Chi-Squared Test

A test for comparing observed to expected frequencies

Summary

1. Read the data into R ;
2. Visualize the data;
3. Choose an appropriate χ^2-test ;
4. Report a conclusion;
5. Incorporating this in scientific literature.

You can download the tutorial here and the required data set here.

2.1 Read the Data into R

Chi-squared tests usually involve a limited number of variables (otherwise you should probably use a binomial GLM instead). We can therefore just enter the data manually:

```
# Example 1: A vector of observed frequencies per category
Observed <- c(rose = 33, tulip = 6, dandelion = 11)

# Example 2: A contingency table
ConTable <- data.frame(
  men   = c(41, 68),
  women = c(33, 82)
```

Table 2.1: First 5 rows of a tidy data set with only categorical data.

smoking	sex
FALSE	man
FALSE	man
FALSE	man
FALSE	man
FALSE	man
...	...

```
)
rownames(ConTable) <- c("non-smoking", "smoking")
```

I have a tidy data file

If you have data in tidy format, you can easily generate a contingency table with the function `table`:

Details

- Save the data in a folder;
- Open RStudio and create a new R markdown file; (**File** > **New File** > **R Markdown**)
- Save your R markdown file to the same location as the data;
- Set working directory to source file location. (**Session** > **Set Working Directory** > **To Source File Location**)

```
# Read the data
DF <- read.csv("binary-data.csv")

# Convert tidy data to a contingency table
ConTable <- table(DF)

# Print the object to confirm it worked
print(ConTable)
```

```
##        sex
## smoking man woman
##   FALSE  41    33
##   TRUE   68    82
```

Common mistakes

Table 2.2: Contingency table of smoking and sex.

	men	women
non-smoking	41	33
smoking	68	82

- Remember to include the file extension (e.g., "binary-data**.csv**") when typing the file name.
- You cannot read Excel files (.xls, .xlsx) with this function. Instead, follow the guide here, or save your data as CSV.
- Don't open the CSV file with Excel. You don't need Excel or Google Sheets or any other program besides R and RStudio. If you have saved your data as CSV, you can close Excel.

2.2 Visualize the Data

If your data can be summarized as a 2×2 contingency table, I recommend just including it as is:

Plotting categorical data

There are several options for plotting categorical data. In summary:

- Marginal bar charts cannot show combinations of variables;
- Grouped bar charts *can* show combinations of variables;
- A mosaic plot is an alternative to a grouped bar chart.

Marginal bar chart

A marginal bar chart shows you the totals of one variable, summed over all other variables. They are what you would find at the *margins* of a contingency table if you were to include the totals:

```
barplot(rowSums(ConTable))
barplot(colSums(ConTable))
```

While this provides easy insight into the marginal distribution of smoking (A) and sex (B), it cannot show you combinations of the two, and is therefore not always informative.

Of course, if your data only contains counts of a single variable, then a marginal bar chart is appropriate:

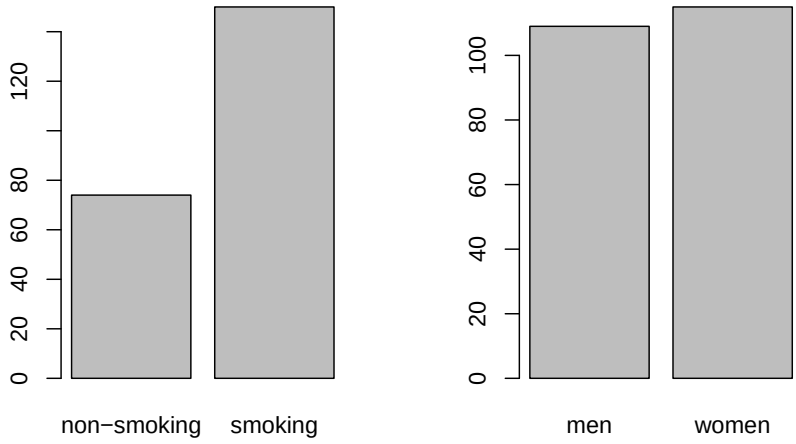
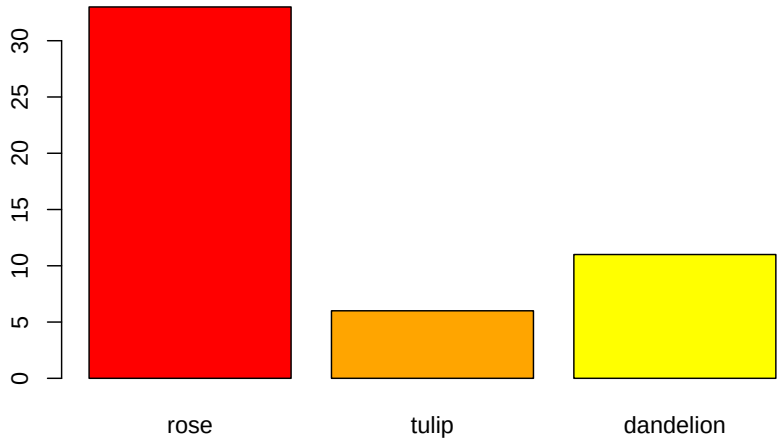


Figure 2.1: Marginal bar chart of smoking (A) and sex (B).

```
barplot(Observed, col = c("red", "orange", "yellow"))
```



Grouped bar chart

A grouped bar chart shows the frequencies of one variable, split by another:

```
barplot(as.matrix(ConTable), beside = TRUE, xlim = c(1, 8), col = 3:4)
legend("topright", legend = rownames(ConTable), pch = 15, col = 3:4)
```

A grouped bar chart has the advantage of showing not the marginals, but combinations of smoking and sex in a single figure.

Improving your plot

Some basic adjustments that improve bar charts:

- Add an informative caption;
- Keep the numbers on the axes horizontal where possible;
- Remove unnecessary plot elements, like boxes around the bars;
- **Never** start a bar chart at a different height than zero. This is fine for just about every other type of plot, but in a bar chart, it warps the perceived difference in relative height.

The changes to the code I made below can all be found in the help pages `?barplot` and `?par`. I am also a big fan of the `eaxis` function from the package `sfsmisc`. The caption is added as a chunk option (i.e., ```` ```{r, fig.cap = "..."} ````).

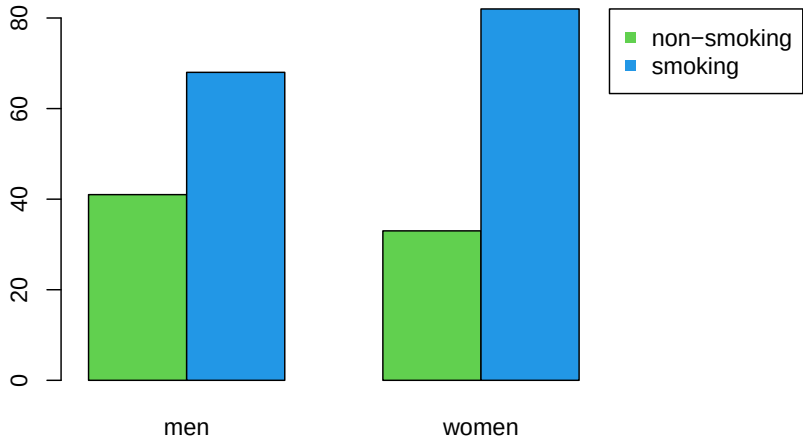


Figure 2.2: Grouped bar chart of smoking and sex.

```
# Change the margins (bottom, left, top, right)
par(mar = c(2.5, 4, 0, 0) + 0.1)

# barplot() expects a matrix. We will need this object several times:
M <- as.matrix(ConTable)

# Color-blind friendly colors picked from: https://colorbrewer2.org/
cols <- c("#1f78b4", "#b2df8a")

# Create the coordinate system
barplot(M, beside = TRUE, xlim = c(0, 9), ylim = c(0, 90), axes = FALSE,
        col = cols, border = NA, xaxs = "i", ylab = "Frequency")

# Add y-axis
eaxis(2)

# Add a lightgrey background
polygon(x = c(-1, 7, 7, -1, -1), y = c(0, 0, 100, 100, 0),
        col = "grey95", border = NA)

# Add a simple grid
```

```
abline(h = seq(0, 100, 10), col = "white", lwd = 1)
abline(h = seq(0, 100, 20), col = "white", lwd = 1.5)

# Redraw the barplot on top
barplot(M, beside = TRUE, xlim = c(1, 8), axes = FALSE, add = TRUE,
        col = cols, border = NA)

# Create a legend
legend("topright", legend = rownames(M), pch = 15, col = cols, bty = "n")

# Restore the default margins for subsequent plots
par(mar = c(5, 4, 4, 3) + 0.1)
```

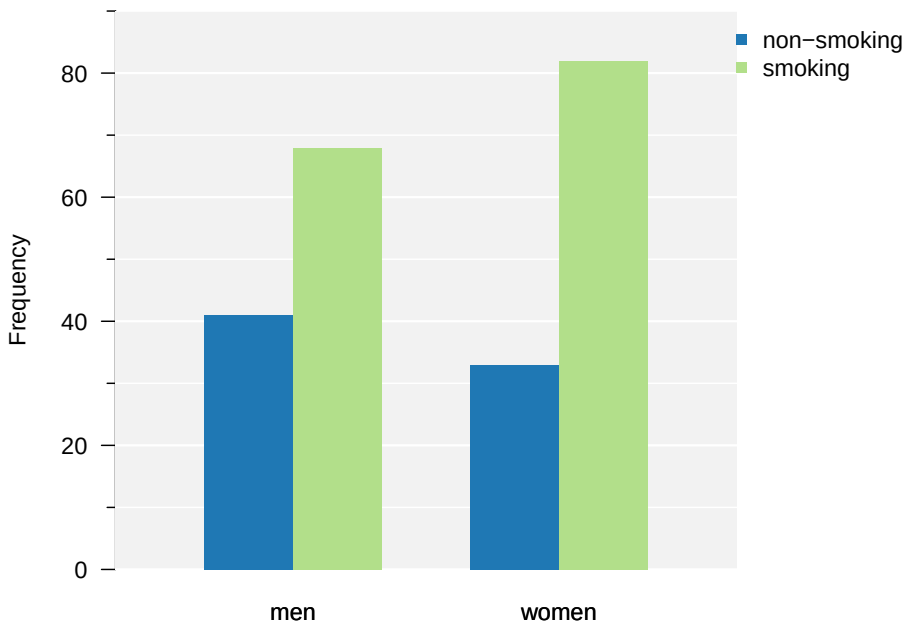


Figure 2.3: Grouped bar chart of smoking and sex.

From the plot we can conclude there are more smokers than non-smokers. Smoking appears slightly more common in females.

Mosaic plot

A mosaic plot is *similar* to a stacked bar chart (groups pasted on top of each other). However, in a mosaic plot, both the height *and* the width of the bars is proportional to the frequency of that group:

```
mosaicplot(ConTable, main = "", col = 3:4)
```

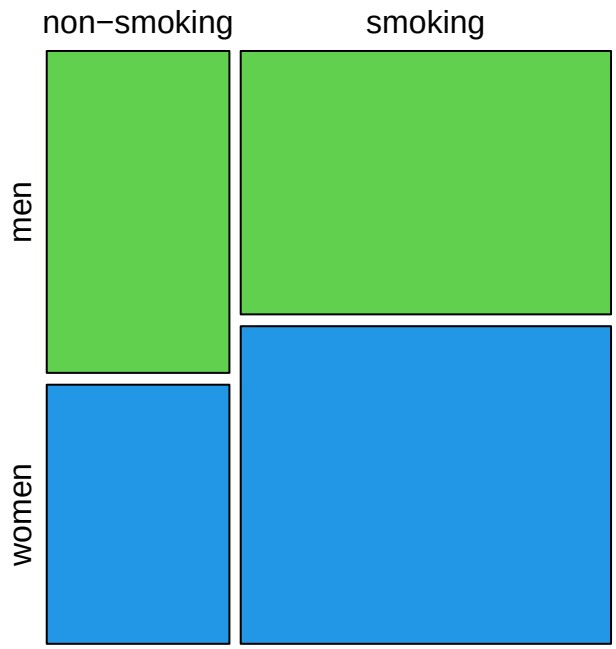


Figure 2.4: Mosaic plot of smoking and sex shows smoking is relatively, slightly more common in women.

Mosaic plots are not very commonly used, but can provide an alternative to bar charts to summarize categorical data effectively, especially with an appropriate color scheme to distinguish between categories.

2.3 Conduct a Chi-Squared Test

If you have a single set of observed values

To compare a single set of observed values to their expected proportions:

```
# Proportions in which the flowers were planted
expected <- c(1/2, 1/4, 1/4)

# Did the flowers sprout in a different ratio than I planted them?
chisq.test(Observed, p = expected)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  Observed
## X-squared = 6.12, df = 2, p-value = 0.04689
```

If you have a contingency table

To conduct a χ^2-test for independence:

```
chisq.test(ConTable)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  ConTable
## X-squared = 1.6293, df = 1, p-value = 0.2018
```

I received a warning

If you receive the following warning:

```
Warning message:
In chisq.test(Observed, p = expected) :
  Chi-squared approximation may be incorrect
```

That means at least one of the expected values is less than 5. The solution is to use Monte Carlo simulation to estimate the p-value (for a vector of observed frequencies) use an exact test (for contingency tables):

```
# Monte Carlo (for a vector of observed frequencies)
chisq.test(Observed, simulate.p.value = TRUE)

# Exact test (for contingency tables)
fisher.test(ConTable)
```

2.4 Correctly Phrase the Results

For a **vector of observed frequencies**:

If the p-value is **less** than the chosen level of significance

(This is the case in the flower example.)

Examples of precise language:

- Flowers species were counted in a ratio significantly different from the $2:1:1$ ratio in which they were planted ($\chi^2 = 24.8$, $p = 4.20 \cdot 10^{-6}$);
- We observed significantly different proportions of flower species than expected ($\chi^2 = 24.8$, $p = 4.20 \cdot 10^{-6}$).

Examples of incorrect, incomplete, or imprecise language:

- The alternative hypothesis was true // The null-hypothesis was false;
- The difference was significant ($p < 0.05$);
- Flowers species sprout in a different ratio than the ratio in which they are planted.

Why paste tense?

The results of a single experiment, no matter how convincing, can never prove something to be true. The results *were* observed, and in this one experiment, A *was* better than B.

Use present tense only for statements that have been demonstrated repeatedly, are generally agreed upon, or are easily observable, e.g.:

- Smoking causes cancer;
- Current climate-change is mainly caused by human activities;
- Most people use smartphones nowadays.

If the p-value is **greater** than the chosen level of significance

Examples of precise language:

- Counts of flowers species did not differ significantly from the $2:1:1$ ratio in which they were planted ($\chi^2 = ...$, $p = ...$);
- There is insufficient evidence to conclude a difference from a $2:1:1$ ratio ($\chi^2 = ...$, $p = ...$);
- There was a trend towards more rose and fewer tulips than expected under a $2:1:1$ ratio, but this difference was not significant ($\chi^2 = ...$, $p = ...$).

Examples of incorrect, incomplete, or imprecise language:

- Planting and sprouting ratios were equal // The $2:1:1$ ratio was correct;
- There was no difference ($p < 0.05$);
- We accept the null-hypothesis.

Why can't I say I *accepted* the null-hypothesis?

This is imprecise language because it distorts the order of null-hypothesis significance testing. Every tests *starts* with pretending the null-hypothesis is true,

and then considering how rare a result this would be. You did not accept the null-hypothesis *because* of the p-value, but rather, you started by taking on the null-hypothesis to even compute that p-value.

For **contingency tables**:

If the p-value is **less** than the chosen level of significance

Examples of precise language:

- Smoking behavior differed significantly by sex ($\chi^2 = ...$, $p = ...$);
- Females were proportionally significantly more likely to be smokers ($\chi^2 = ...$, $p = ...$).
- Smoking behavior and sex were significantly dependent ($\chi^2 = ...$, $p = ...$).

Examples of incorrect, incomplete, or imprecise language:

- The alternative hypothesis was true // The null-hypothesis was false;
- The difference was significant ($p < 0.05$);
- Sex and smoking are independent of each other.

Why paste tense?

The results of a single experiment, no matter how convincing, can never prove something to be true. The results *were* observed, and in this one experiment, A *was* better than B.

Use present tense only for statements that have been demonstrated repeatedly, are generally agreed upon, or are easily observable, e.g.:

- Smoking causes cancer;
- Current climate-change is mainly caused by human activities;
- Most people use smartphones nowadays.

If the p-value is **greater** than the chosen level of significance

Examples of precise language:

- Smoking behavior did not differ significantly by sex ($\chi^2 = ...$, $p = ...$);
- There is insufficient evidence to conclude a difference in smoking behavior between men and women ($\chi^2 = ...$, $p = ...$);
- Smoking behavior and sex did not depend significantly on each other (χ^2, p).

Examples of incorrect, incomplete, or imprecise language:

- Smoking behavior was equal for men and women // Sex and smoking were independent;
- There was no difference ($p < 0.05$);
- We accept the null-hypothesis.

Why can't I say I *accepted* the null-hypothesis?

This is imprecise language because it distorts the order of null-hypothesis significance testing. Every tests *starts* with pretending the null-hypothesis is true, and then considering how rare a result this would be. You did not accept the null-hypothesis *because* of the p-value, but rather, you started by taking on the null-hypothesis to even compute that p-value.

2.5 Incorporating This in a Paper

Here you can find examples of how to justify the methods used, explain the results found and write a discussion. This is meant to show you what belongs where, and what level of detail is common in scientific literature.

Remember to use your own words—**paraphrase to avoid plagiarism**.

Methods

(In this section, you should describe the data collection and analysis to an extent that a fellow expert could reproduce your study.)

All 226 employees at Example company were inquired about their sex and smoking behavior in November 2022. A total of $n = 224$ employees responded.

All statistical analyses were conducted in R, version 4.2.1, using the RStudio interface.[R Core Team, 2022, RStudio Team, 2022] Observed and expected frequencies were compared with a Pearson's chi-squared test.

Note:

- Check your R version number with `version`;
- Justify the type of test used; *(If you use a χ^2-test, all expected (not observed) counts should be greater than 5.)*
- You should cite any packages used to perform analyses, generate tables or figures that made it into the paper, e.g.;
  - Figures were created with the `ggplot2` package.[Wickham, 2016] *(Not the case in the tutorial.)*
- You should *not* cite packages used only internally, e.g.:
  - We used `readxl` to enter the data in R.[Wickham and Bryan, 2022]

- It is important to mention adequately how the data was collected. For survey research like that in the example, the respondence rate can have a large influence on the conclusions: What if smoking females are less likely to respond?

Results

(In this section you should mention the results without giving any interpretation yet.)

For general advice on phrasing, see: Correctly Phrase the Results. In this section, you should include your table and/or figure of the data and explain in brief what the outcome of the χ^2-test was.

It is very common to see bar charts with significance stars, but the interpretation of these stars is not consistent and you should mention the actual, non-discretized p-value at least somewhere in the results.

Discussion

(In this section, you should not mention results, but the conclusions based on those results.)

For a **vector of observed frequencies**:

If the test was significant **and** the difference large enough to be biologically relevant

(This is the case in the flower example.)

- Roses appear to sprout more often, and tulips less often than dandelions, though more replicates of the experiment are necessary for a conclusive answer;
- Differences in seed resilience of roses, tulips and dandelions may cause them to sprout disproportional to their planting ratios.
- *(Overclaiming)* Roses are better at sprouting than tulips or dandelions.
- *(Overgeneralizing)* Thorned flowers appear to sprout more often than flowers without thorns.

If the test was insignificant **or** the difference too small to be biologically relevant:

- *(If the study was sufficiently powerful)* Despite a fairly large sample size, we were unable to demonstrate a difference in planting and sprouting ratios, suggesting these species are equally likely to sprout when planted.
- *(If the study was underpowered)* We were unable to demonstrate a difference in planting and sprouting ratios, though a multiple replications of the experiment in different fields may yield a more conclusive answer.
- *(Appeal to ignorance)* We have demonstrated there is no difference in planting and sprouting ratios.

Why can't I conclude there is no difference?

This is an inherent limitation of the p-value. Under the null-hypothesis, any p-value is as likely as any other (uniformity). Therefore, even if the p-value is very large, this cannot be interpreted as evidence *for* the null-hypothesis.

For example, a p-value of 0.20 is just as common a result as 0.80 when the null-hypothesis is correct. The p-value can *only* be used as evidence *against* the null-hypothesis (when it is small).

If you want to perform a test to find out whether the null-hypothesis is correct, what you are looking for is called an equivalence test.

For **contingency tables**:

If the test was significant **and** the difference large enough to be biologically relevant

- Women may be more inclined to smoke at work, though more replicates of the experiment at different companies are necessary for a conclusive answer;
- Anti-smoking campaigns at Example company may have been relatively more effective at targeting men, though a longitudinal intervention study could provide a more conclusive answer.
- *(Overclaiming)* Anti-smoking campaigns at Example company have been more successful in targeting men.
- *(Overgeneralizing)* Men are less inclined to smoke at work.

If the test was insignificant **or** the difference too small to be biologically relevant:

(This is the case in the smoking example.)

- *(If the study was sufficiently powerful)* Smoking remains prevalent at Example company, irrespective of sex.
- *(If the study was underpowered)* We were unable to demonstrate a difference in smoking ratios of men and women at Example company.
- *(Appeal to ignorance)* We have demonstrated smoking and sex are independent at Example company.

Why can't I conclude there is no difference?

This is an inherent limitation of the p-value. Under the null-hypothesis, any p-value is as likely as any other (uniformity). Therefore, even if the p-value is very large, this cannot be interpreted as evidence *for* the null-hypothesis.

For example, a p-value of 0.20 is just as common a result as 0.80 when the null-hypothesis is correct. The p-value can *only* be used as evidence *against* the null-hypothesis (when it is small).

If you want to perform a test to find out whether the null-hypothesis is correct, what you are looking for is called an equivalence test.

General note: Counts from a single location, work place, or petri dish are never convincing enough for strong conclusive remarks. It is better to have multiple replicates of the whole experiment (fields, companies, petri dishes) and analyze the results with, for instance, a binomial GLM.

Generating citations

Any packages used in analysis should be cited in the methods section. The citations used here can be easily obtained as follows (note that this requires the packages to be installed):

```
citation()          # Use this to cite R
RStudio.Version()   # Use this to cite RStudio
citation("ggplot2") # Use this to cite a package (here: ggplot2)
```

You can add these entries to a reference manager (e.g., Zotero), or keep a BibTeX file yourself, as shown here.

Chapter 3

ANOVA

A method test to compare any number of means to see if they differ significantly. Observations are assumed to be **independent**, deviations from the group means are assumed to follow a **normal distribution**, and groups are assumed to have **equal variance**.

Summary

1. Read the data into R ;
2. Visualize the data;
3. Fit an ANOVA:
    - Perform visual diagnostics to look for deviations from the assumptions;
    - Perform an omnibus test if the assumptions appear reasonable;
    - Perform a post-hoc test if the omnibus test was significant;
4. Report a conclusion;
5. Incorporating this in scientific literature.

You can download the tutorial here and the required data set here.

3.1 Read the Data into R

I recommend saving data as comma-separated values (CSV). If you prefer reading data directly from Excel, have a look here.

Details

- Save the data in a folder;
- Open RStudio and create a new R markdown file; (**File** > **New File** > **R Markdown**)
- Save your R markdown file to the same location as the data;
- Set working directory to source file location. (**Session** > **Set Working Directory** > **To Source File Location**)

```
DF <- read.csv("three-groups.csv")
```

Did that work? There are several ways to check:

```
str(DF)
summary(DF)
head(DF)
```

Explanation of the output

```
str(DF)
```

```
## 'data.frame':    90 obs. of  4 variables:
##  $ SBP      : int  122 119 120 129 127 118 120 124 126 117 ...
##  $ treatment: chr  "thiazide" "thiazide" "thiazide" "thiazide" ...
##  $ age      : int  32 34 36 41 35 38 36 33 39 32 ...
##  $ sex      : chr  "female" "female" "female" "male" ...
```

This command shows the structure of the data. It is a data frame with $n = 60$ observations of 4 variables. `SBP` and age are stored as integers (`int`), while `treatment` and `sex` are stored character vectors (`chr`).

```
##       SBP         treatment              age            sex
##  Min.   :114.0   Length:90          Min.   :23.00   Length:90
##  1st Qu.:122.0   Class :character   1st Qu.:32.00   Class :character
##  Median :127.0   Mode  :character   Median :35.00   Mode  :character
##  Mean   :131.5                      Mean   :35.69
##  3rd Qu.:142.0                      3rd Qu.:38.00
##  Max.   :201.0                      Max.   :83.00
```

```
##   SBP treatment age    sex
## 1 122  thiazide  32 female
## 2 119  thiazide  34 female
## 3 120  thiazide  36 female
## 4 129  thiazide  41   male
## 5 127  thiazide  35 female
## 6 118  thiazide  38 female
```

My output looks different

Then provided you did everything else correctly, the most likely reason is that your data was saved with a version of Excel where a comma is used as a decimal separator (e.g., the Dutch version). The solutions for this is simple, use `read.csv2`:

```
DF <- read.csv2("some-data-with-commas.csv")
```

If that doesn't work either, go over the details shown above, or watch the instructions in the video.

Common mistakes

- Remember to include the file extension (e.g., "two-groups**.csv**") when typing the file name.
- You cannot read Excel files (.xls, .xlsx) with this function. Instead, follow the guide here, or save your data as CSV.
- Don't open the CSV file with Excel. You don't need Excel or Google Sheets or any other program besides R and RStudio. If you have saved your data as CSV, you can close Excel.

The next step is to ensure categorical variables are read as factors. This will allow us to use generic functions like `plot` and `summary` in a more meaningful way.

Why not just use `character`?

A character vector is just strings of text, numbers and/or symbols. If you were to produce a `summary`, this happens:

```
summary(DF$treatment)
```

```
##    Length     Class      Mode
##        90 character character
```

It tells us this object contains 60 values, and it is stored and treated as a string of text.

The generic `plot` function doesn't even work at all:

```
plot(SBP ~ treatment, DF)
```

```
Error in plot.window(...) : need finite 'xlim' values
In addition: Warning messages:
1: In xy.coords(x, y, xlabel, ylabel, log) : NAs introduced by coercion
2: In min(x) : no non-missing arguments to min; returning Inf
3: In max(x) : no non-missing arguments to max; returning -Inf
```

Now convert the variables to factors and see how that changes the output. *(Run the code below, then run* `summary(DF$treatment)`*, or* `plot(DF$treatment)` *for example.)*

```
DF$treatment <- factor(DF$treatment)
DF$sex       <- factor(DF$sex)
```

3.2 Visualize the Data

3.3 Fit an ANOVA

3.4 Correctly Phrase the Results

3.5 Incorporating This in a Paper

Chapter 4

Linear Model

...

...

Chapter 5

Generalized Linear Model

...

...

Chapter 6

Mixed Model

...

...

Chapter 7

GLMM

...

...

Chapter 8

Exploration: PCA

A model for dependent data. The dependency can occur due to repeated measures, spatial correlation, hierarchy, or nesting.

Summary

(…) under construction.

8.1 Read the Data into R

(…) under construction.

Chapter 9

Reading & Storing Data

This chapter provides some advice for how to store data and shows you various ways in which you can read it.

Summary

- **Store data in tidy format**. There are exceptions to this format, but only one for the methods listed on this website; (namely, long-format for time series)
- **Separate data *entry* and *analysis***. Excel invites you to do questionable things, like adding figures and explanation to raw data. Don't do this. Never edit or clutter raw data. If you must use Excel for data exploration, create a separate file for this purpose.
- **Use short, informative variable names**, using only lower- and uppercase letters, numbers,[1] and if needed periods (.) or underscores (_). Do not add the unit of measurement, or any explanation. This belongs in the meta data. If variable names are still long, consider using abbreviations. Avoid special characters (!@#$%^&*-a=+,<>?/).
- **Keep a meta data file**, or sheet, that explains variables, abbreviations, unit of measurement, etc. See here for an example. This also helps keep variable names short and free of special characters.
- **Save data as comma-separated values** (.csv), rather than as an Excel file (.xlsx). Saving into this format automatically removes any content statistical software cannot read anyway, like figures and coloring. This file format is suitable for most sizes of data commonly encountered in the life sciences (anything that still fits into RAM on an ordinary computer).

[1] When using numbers, keep in mind that R does not accept variable names *starting* with a number. R will automatically convert these.

Table 9.1: The first 5 rows of the <tt>iris</tt> data set. Each row is a flower and each column a property.

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
...	...	...	...	...

9.1 Tidy Format

Collect data in a simple, consistent format called *tidy data*,[Wickham, 2014] such that minimal effort is required to clean the data once you get to the analysis: Rows represent observations, columns represent the variables measured for those observations:

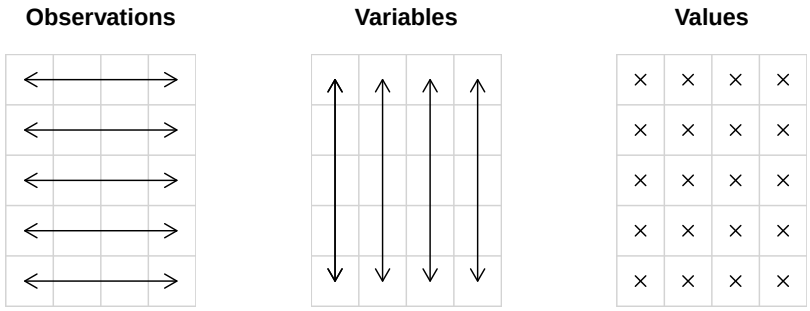


Figure 9.1: The basic principle of tidy data. This data set has 5 observations of 4 variables.

Good example

The `iris` data set (preinstalled with R) is in tidy format:

Here, the rows each represent one observation (a distinct flower), and the columns represent the variables measured/recorded (physical dimensions and species).

Bad example

A common deviation from tidy format is to represent groups as columns:

Table 9.2: Systolic blood pressure (SBP) of 10 individuals. (untidy)

women	men
114	123
121	117
125	117
108	117
122	116

Table 9.3: Sex and systolic blood pressure (SBP) of 10 individuals. (tidy)

sex	SBP
female	114
female	121
female	125
female	108
female	122
male	123
male	117
male	117
male	117
male	116

Do you have different groups? Time points? Replicates of the experiment? Then try to adhere to the same principle: Columns are variables. Simply add a variable that indicates which group/time point/replicate this observation belongs to:

Converting to tidy format

If you have data split by group/time point/replicate here is how you can convert it to tidy format:

```
# The untidy data set
Untidy
```

```
##   women men
## 1   114 123
## 2   121 117
## 3   125 117
## 4   108 117
```

```
## 5   122 116
```

```
# Convert by hand
Tidy <- data.frame(
  sex = rep(c("female", "male"), each = nrow(Untidy)),
  SBP = c(Untidy$women, Untidy$men)
)

# Convert using a package (install if missing)
library("reshape2")
melt(Untidy)
```

```
##    variable value
## 1     women   114
## 2     women   121
## 3     women   125
## 4     women   108
## 5     women   122
## 6       men   123
## 7       men   117
## 8       men   117
## 9       men   117
## 10      men   116
```

If you have a simple data set like the one shown here, converting with the package `reshape2` is easiest. Converting by hand may be slightly more work, but I prefer it because you can easily see what's going on, add more variables if needed, etc.

9.2 Separate Data Entry & Analysis

Spreadsheet software like Excel or Google Sheets is great for easy data entry. A well-designed spreadsheet can even partially protect against errors, by limiting what values are allowed in certain columns (e.g., how `Species` is restricted here).[2]

Beware 'intelligent' conversions

Programming requires exact precision in syntax, which can be very frustrating for beginners. To address this, many programs try to intelligently interpret your input:

While this usually a good thing, Excel does many such corrections without notice, and some even irreversibly. This can have serious repercussions for

[2] In Google Sheets, you can do this by going to **Data** > **Data validation**.

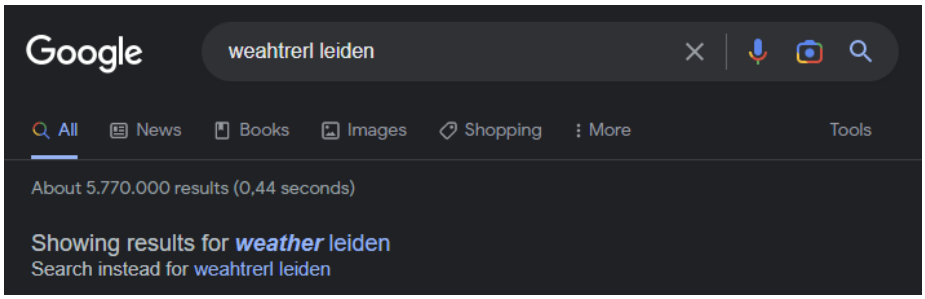


Figure 9.2: My hasty attempt to figure out which clothes to wear to work.

scientific research. For example, a 2016 article found widespread errors in gene names due to autocorrected Excel input.[Ziemann et al., 2016] Always check whether your input is correctly interpreted, or change the data type to "text" instead of "general" for everything you enter.

Once the data are correctly entered into some spreadsheet software (Excel, Google Sheets), this is where the data entry ends. Do not add any markup, coloring, notes, figures, or tables.

As a general principle, **raw data should never be edited**, nor should it be cluttered with additional information. This does not belong in your raw data, but either in a meta data file, or in some separate file for analysis that takes the raw data file as input.

Some additional guidelines for tidy data entry include:

- Avoid empty rows or columns for spacing, start at row 1, column 1;
- Avoid the 'merge cells' option, this breaks the row/column format;
- Limit a spreadsheet to one data set. Adding multiple data sets to one sheet may look like you have more oversight in *Excel*, but it makes any downstream analysis more convoluted.

Excel Is Not Statistical Software

Even if you do create a separate file to explore/analyze your data, I still advice against using Excel for this, for the following reasons:

A history of inaccuracies

Excel has long been a program ridden with computational inaccuracies and slow to correct its mistakes.

Non-trivial errors in the calculation of discrete probability distributions, as well as severe limitations of its random number generator (`RAND()`) were found in Excel 1997,[Knüsel, 1998, McCullough and Wilson, 1999] exacerbated in Excel

2000/XP,[McCullough and Wilson, 2002] and only partially addressed in Excel 2003, while simultaneously introducing new problems.[McCullough and Wilson, 2005, Knüsel, 2005] Perhaps unsurprisingly then, Excel 2007 still failed some of the same accuracy tests in use since 1998.[McCullough and Heiser, 2008]

By the time Excel 2010 was released, persisting issues with the computation of probabilities and quantiles had been addressed, and the `RAND()` function had improved.[Mélard, 2014] Nevertheless, the authors of a paper reviewing Excel 2010 concluded:

> [W]ithout a prompt reaction from Microsoft, the already very critical view against using spreadsheets for doing any statistical analysis […] will be even more justified. **It is standard practice for software developers to fix errors quickly and correctly. Microsoft has fixed the errors in the statistical procedures of Excel neither quickly nor correctly.** The recent improvements reported in this paper should not hide the fact that Microsoft is still marketing a product that contains known errors.

I have emphasized the part that summarizes this section. Excel is proprietary software from a developer that cares little for its statistical accuracy. In contrast, R, Python, Julia and the likes are free & open source (meaning anyone can view the source code).

Few new articles have since been published on this issue—presumably because R and Python have since become the *de facto* industry standard for statistical analysis.

Lack of serious modeling options

If statistics were a toolbox, then Excel would be a hammer. Yes, you can do a lot with it, but if you're trying to cut something in half, or insert a screw, maybe use a different tool. Excel has add-ins, but these are neither open source, nor curated.

As of writing, R has 18,834 free, open source packages, offering anything from statistical tests, models, machine learning, visualization (`base`, `ggplot2`, `plotly`), literate programming (`rmarkdown`, `quarto`), presentations (e.g., `xaringan`), paper writing (`rticles`), book authoring (`bookdown`), app and dashboard development (`shiny`), and even interfaces to state-of-the-art deep learning frameworks (e.g., `tensorflow`, `keras`).

Computations are opaque

Analyses performed in Excel consist of cells with formulas referring to other (ranges of) cells. This is not a problem for simple operations, like adding or subtracting two columns. But in actual analysis, there will be many such steps, and Excel 'code' becomes much harder to read than other programming languages, since you can only view the contents of one cell at a time.

The worst offenders are large Excel sheets with both data and computations, requiring not only that you trace back the operations performed in multiple cells, but also that you scroll between (often distant) locations in the sheet. Bugs *will* appear in large analyses and it is needlessly difficult to trace the origin of mistakes.

Excel is also slow, especially in files combining data and analysis; it is paid software, requiring a license; and its 'arrange cells however you see fit' approach, makes analyses much harder to reproduce.

If you want an easy interface to enter your data, Excel is great software. Any elaborate analyses in Excel though, are inefficient at best and severely limited at worst.

9.3 Meta Data

To help keep variable names short, to include the unit of measurement, to explain how something was measured, or to provide other relevant comments about the data, you can keep a meta data file (or sheet). An example can be found here.

Meta data means *data about data*. If you take a picture with your phone, meta data contains information about how the picture was taken (shutter time, ISO, etc.), or where the picture was taken. Naturally, you do not want this information pasted onto the picture itself. In a similar vein, you should not include comments about the data, in the raw data itself. This is why you should keep a separate file for meta data.

9.4 Reading Data as CSV

Why this format?

Saving into CSV format automatically removes any content statistical software cannot read anyway, like figures and coloring. This file format is suitable for most sizes of data commonly encountered in the life sciences (anything that still fits into RAM on an ordinary computer).

For very large data sets, there are special high-performance file types, like HDF5 or parquet. However, these offer little to no benefit for the small-sized data sets used in the tutorials here.

Setting up your working directory

- Save the data in a folder;
- Open RStudio and create a new R markdown file; (**File** > **New File** > **R Markdown**)

- Save your R markdown file to the same location as the data;
- Set working directory to source file location. (**Session > Set Working Directory > To Source File Location**)

Provided your working directory is set up correctly, you can read CSV files as follows:

```
DF <- read.csv("NameOfYourFile.csv")
```

Here, `DF` is an arbitrary name. I use it because it is short (you will be typing this name a lot) and because it is an abbreviation of 'data frame', so it keeps code easy to read for others and your future self.

To check that your data has been read correctly by the software, try one of the following:

```
str(DF)
summary(DF)
head(DF)
tail(DF)
```

What it should look like

The `str` function displays the structure of the object. In case your file is in tidy format and read correctly, it will look something like this:

```
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 .
```

More explanation

The first line tells us this object is a `data.frame` consisting of 150 observations (rows) and 5 variables (columns). It then tells us what kind of information those variables contain:

- `num`: Numeric (any number, stored as a floating point)
- `int`: Integer (whole number)
- `chr`: Character string
- `Factor`: Categorical variable with a fixed number of levels

- Some less common ones.

In the example shown here, the data has been read correctly, because numbers show as `num` and the flower species shows as `Factor`. If your factor shows up as a character (`chr`), you can convert it as follows:

```
DF$x <- factor(DF$x)
```

Where `x` is the name of the variable you want to convert to factor.

The `summary` shows some summary statistics for numeric variables, and counts of categories for factors:

```
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

More explanation

For numeric variables, the minimum, first 25%, first 50% (median), the mean (average), first 75% and the maximum are shown. This provides some basic indication of the distribution of the data, and it may point to potential outliers if the minimum or maximum are unrealistically far away.

For factors, the counts of categories are useful to see whether the data are balanced (same number of observations per category).

Finally, `head` and `tail` show the first and last rows of the data, respectively (6 rows by default):

```
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
tail(iris)
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 145          6.7         3.3          5.7         2.5 virginica
## 146          6.7         3.0          5.2         2.3 virginica
## 147          6.3         2.5          5.0         1.9 virginica
## 148          6.5         3.0          5.2         2.0 virginica
## 149          6.2         3.4          5.4         2.3 virginica
## 150          5.9         3.0          5.1         1.8 virginica
```

My output looks different

First, make sure your data are in tidy format. That also means no empty rows before your data starts, no comments in cells where there is no data, etc.

If you're sure your data is in the right format, another very common issue is the **decimal separator**. In some countries (including The Netherlands), decimal numbers are separated with a comma, e.g.:

$$\begin{aligned} \text{Dutch style:} \quad & 3 - 0{,}1 = 2{,}9 \\ \text{English style:} \quad & 3 - 0.1 = 2.9 \end{aligned}$$

If you have a Dutch version of Excel, then CSV files are actually not separated by commas but semi-colons, because the software would otherwise not be able to tell new numbers and decimal numbers apart.

The solution for this is fortunately very simple, use read.csv2:

```
DF <- read.csv2("NameOfYourFile.csv")
```

If this too does not solve your problem, try copying the entire error message and searching for it online. Usually, there will have been someone else with the same problem on StackOverflow, so you can try the solutions offered there.

9.5 Reading Data from Excel

Reading data directly from Excel only works if R 'understands' the file. An Excel file cluttered with figures or special functionality will not be readable, which is why I recommend just using CSV instead. Despite the drawbacks, experience shows that many people prefer to read data directly from Excel anyway, so I will demonstrate this too.

Setting up your working directory

- Save the data in a folder;
- Open RStudio and create a new R markdown file; (**File** > **New File** > **R Markdown**)
- Save your R markdown file to the same location as the data;
- Set working directory to source file location. (**Session** > **Set Working Directory** > **To Source File Location**)

Provided your working directory is set up correctly, there are many ways to directly read your Excel file into R, for example:

```
# Using readxl
library("readxl")
DF <- read_xlsx("NameOfYourFile.xlsx", sheet = 1)

# Using xlsx
library("xlsx")
DF <- read.xlsx("NameOfYourFile.xlsx", sheetIndex = 1)

# Using fread
library("data.table")
DF <- fread(file = "NameOfYourFile.xlsx")
```

If you are already familiar with the syntax of one of these packages, there is no need to switch. If people send me Excel sheets, I usually use `readxl`. A more general and high-performance function is `fread` from the package `data.table`.

References

Bibliography

John Fox and Sanford Weisberg. *An R Companion to Applied Regression*. Sage, Thousand Oaks CA, third edition, 2019. URL https://socialsciences.mcmaster.ca/jfox/Books/Companion/.

Leo Knüsel. On the accuracy of statistical distributions in microsoft excel 97. *Computational Statistics & Data Analysis*, 26(3):375–377, January 1998. doi: 10.1016/s0167-9473(97)81756-2. URL https://doi.org/10.1016/s0167-9473(97)81756-2.

Leo Knüsel. On the accuracy of statistical distributions in microsoft excel 2003. *Computational Statistics & Data Analysis*, 48(3):445–449, March 2005. doi: 10.1016/j.csda.2004.02.008. URL https://doi.org/10.1016/j.csda.2004.02.008.

B D McCullough and Berry Wilson. On the accuracy of statistical procedures in microsoft excel 2000 and excel xp. *Computational Statistics & Data Analysis*, 40(4):713–721, 2002.

B.D. McCullough and David A. Heiser. On the accuracy of statistical procedures in microsoft excel 2007. *Computational Statistics & Data Analysis*, 52(10):4570–4578, June 2008. doi: 10.1016/j.csda.2008.03.004. URL https://doi.org/10.1016/j.csda.2008.03.004.

B.D. McCullough and Berry Wilson. On the accuracy of statistical procedures in microsoft excel 97. *Computational Statistics & Data Analysis*, 31(1):27–37, July 1999. doi: 10.1016/s0167-9473(99)00004-3. URL https://doi.org/10.1016/s0167-9473(99)00004-3.

B.D. McCullough and Berry Wilson. On the accuracy of statistical procedures in microsoft excel 2003. *Computational Statistics & Data Analysis*, 49(4):1244–1252, June 2005. doi: 10.1016/j.csda.2004.06.016. URL https://doi.org/10.1016/j.csda.2004.06.016.

Guy Mélard. On the accuracy of statistical procedures in microsoft excel 2010. *Computational Statistics*, 29(5):1095–1128, April 2014. doi: 10.1007/s00180-014-0482-5. URL https://doi.org/10.1007/s00180-014-0482-5.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2022. URL https://www.R-project.org/.

RStudio Team. *RStudio: Integrated Development Environment for R*. RStudio, PBC, Boston, MA, 2022. URL http://www.rstudio.com/.

Hadley Wickham. Tidy data. *Journal of Statistical Software*, 59(10), 2014. doi: 10.18637/jss.v059.i10. URL https://doi.org/10.18637/jss.v059.i10.

Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org.

Hadley Wickham and Jennifer Bryan. *readxl: Read Excel Files*, 2022. URL https://CRAN.R-project.org/package=readxl. R package version 1.4.0.

Mark Ziemann, Yotam Eren, and Assam El-Osta. Gene name errors are widespread in the scientific literature. *Genome Biology*, 17(1), August 2016. doi: 10.1186/s13059-016-1044-7. URL https://doi.org/10.1186/s13059-016-1044-7.