Descriptive statistics

Eva Freyhult, Olga Dethlefsen

Table of contents

Preface

Descriptive statistics is a term describing simple analyses of data that help getting to know the data by describing the data, showing the data and summarizing the data. Descriptive statistics is used to guide down-stream data analysis and often helps to uncover patterns in the data.

Learning outcomes

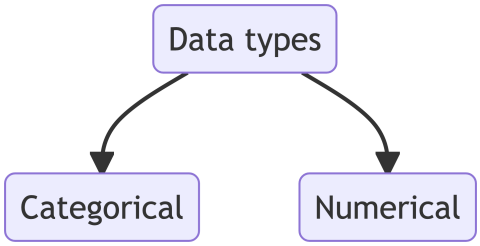
- understand why we are doing descriptive statistics
- understand the difference between data types and be able to select and use appropriate data summaries and plots for each data type
- compute measures of location, including mean and median
- compute measures of spread, including quantiles, variance and standard deviation
- compute population mean and variance
- compute sample mean and variance

Do you see a mistake or a typo? We would be grateful if you let us know via edu.ml-biostats@nbis.se

This repository contains teaching and learning materials prepared and used during "Introduction to biostatistics and machine learning" course, organized by NBIS, National Bioinformatics Infrastructure Sweden. The course is open for PhD students, postdoctoral researcher and other employees within Swedish universities. The materials are geared towards life scientists wanting to be able to understand and use basic statistical and machine learning methods. More about the course https://nbisweden.github.io/workshop-mlbiostatistics/

1 Data types

Data can be divided into different types. We differentiate between categorical (qualitative) and numerical (quantitative) data types.



Categorical data types are further divided into:

- **Nominal**: named, categories are mutually exclusive and unordered
  - *e.g.dead/alive, healthy/sick, WT/mutant, blood group (A/B/ABO/O), male/female, red/green/blue*
- **Ordinal:** named and ordered, categories are mutually exclusive and ordered
  - *e.g. pain (weak, moderate, severe), AA/Aa/aa, very young/young/middle age/old/very old, grade I, II, III, IV*

Numerical data types are further divided into:

- **Discrete**: finite or countable infinite values
  - *e.g. days sick last year, number of cells, number of reads*
- **Continuous**: infinitely many uncountable values

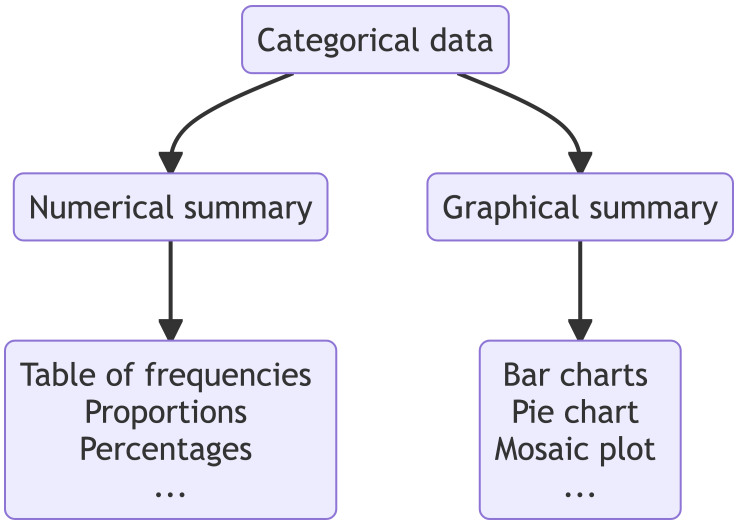
- *e.g. height, weight, concentration*

> **!** Important
>
> Depending on the data type we use different methods to describe, summarize and visualize the data. Beyond descriptive statistics, we even use different methods to analyse the data.

2 Categorical data

```
# load libraries
library(tidyverse)
library(kableExtra)
library(ggplot2)
```

Categorical data can be summarized by counting the number of observations of each category and summarizing in a frequency table or graphically in a bar chart. Alternatively we can calculate the the proportions (or percentages) of each category.



Example 2.1 (Lab mice: sex)**.** Imagine, we run an experiment, in which we follow 100 mice over a period of 24 weeks. To begin with, we record mice sex (male/female). In addition, every week

Table 2.1: Preview of the mice data

id	sex	male	week	weight
1	male	1	5	19.05
1	male	1	6	19.99
1	male	1	7	20.79
1	male	1	8	21.37
1	male	1	9	22.08
1	male	1	10	22.72

we record the weight (g) of each mouse, starting at week 5. How can we summarize the sex (male/female) variable?

The data can be loaded from mice.csv and a preview is shown below.

```
# read in data
mice <- read_csv("data/mice.csv")
mice <- mice %>%
  mutate(weight = round(weight,2))

# # preview data
# mice %>%
#   datatable() %>%
#   formatSignif(columns = c("weight"), digits = 4)

# preview data
head(mice) %>%
  kable() %>%
  kable_styling(full_width = FALSE)
```

Let's focus on only subset of data, the first 10 mice.

```
# select first 10 mice at week 5
mice.10 <- mice %>%
  filter(week == 5) %>%
  filter(id %in% 1:10)

# preview data
head(mice.10) %>%
```

Table 2.2: Sex and weigth data for a subset of 10 mice observed at week 5

id	sex	male	week	weight
1	male	1	5	19.05
2	male	1	5	20.67
3	female	0	5	18.18
4	male	1	5	20.33
5	male	1	5	21.02
6	male	1	5	16.88

```
  kable() %>%
  kable_styling(full_width = FALSE)
```

Clearly, information about male/female falls under categorical data type. Things we can ask here to summarize the data are: how many mice of each category we have, i.e. how many males and how many females and what are the males/females percentages (or proportions). We can also visualize these descriptive statistics in a bar chart of a pie chart.

2.1 Frequency table

```
# count frequencies, percentages and proportions
table.summary <- mice.10 %>%
  group_by(sex) %>%
  tally() %>%
  mutate("percent (%)" = n/sum(n)*100) %>%
  mutate("proportion" = n/sum(n))

# show table
kable(table.summary) %>% kable_styling(full_width = TRUE)
```

Table 2.3: **?(caption)**

(a)

sex	n	percent (%)	proportion
female	3	30	0.3
male	7	70	0.7

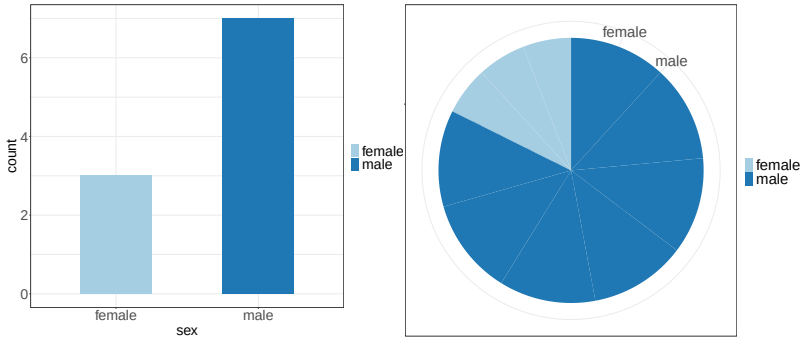
2.2 Bar chart & pie chart

To visualize the frequencies (or percentages or proportions) we can use bar charts referred to as barplots in R.

```
font.size <- 30
my.ggtheme <- theme(axis.title = element_text(size = font.size),
         axis.text = element_text(size = font.size),
         legend.text = element_text(size = font.size),
         legend.title = element_blank())

# use ggplot to draw a bar chart
mice.10 %>%
  ggplot(aes(x = sex, fill = sex)) +
  geom_bar(width = 0.5) +
  scale_fill_brewer(palette = "Paired") +
  theme_bw() +
  my.ggtheme

# draw pie chart
mice.10 %>%
  ggplot(aes(x="", y = sex, fill = sex)) +
  geom_bar(width = 1, stat = "identity") +
  theme_bw() +
  coord_polar("y", start=0) +
  scale_fill_brewer(palette="Paired") +
  xlab("") +
  ylab("") +
  my.ggtheme
```



(a) Bar chart (b) Pie chart

Figure 2.1: Graphical summaries of number and percentage of mice for each observed sex category (male/female)

Example 2.2 (Left-handedness)**.** We are interested in whether left-handedness is associated with suffering from migraine. We collect data on handedness in 30 patients suffering from migraine on regular basis and 40 healthy controls.

The preview of data:

```
# in fact we just generate some random data on handedness (L/H) and migrane (Yes/No)
set.seed(1123) # set random seed
patients <- sample(c("L", "R"), 30, prob=c(0.3, 0.7), replace=TRUE)
controls <- sample(c("L", "R"), 40, prob=c(0.1, 0.9), replace=TRUE)
data.handedness <- rbind(data.frame(group="patient", handedness=patients),
                         data.frame(group="control", handedness=controls)) %>%
  rownames_to_column("id")

# preview data on handedness and migraine
glimpse(data.handedness)
```

```
Rows: 70
Columns: 3
$ id         <chr> "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "1~
$ group      <chr> "patient", "patient", "patient", "patient", "patient", "pat~
$ handedness <chr> "L", "L", "L", "R", "L", "R", "R", "L", "L", "R", "R", "R",~
```

```r
# preview first few observations
head(data.handedness)
```

```
  id   group handedness
1  1 patient          L
2  2 patient          L
3  3 patient          L
4  4 patient          R
5  5 patient          L
6  6 patient          R
```

```r
# preview last few observations
tail(data.handedness)
```

```
   id   group handedness
65 65 control          R
66 66 control          R
67 67 control          R
68 68 control          R
69 69 control          R
70 70 control          R
```

2.3 Summary table: 2 categorical variables

```r
# count number (and %) of left-handed by group (patients / controls)
data.handedness %>%
  group_by(group) %>%
  dplyr::summarize(Total=n(), `Left-handed` = sum(handedness=="L")) %>%
  mutate(`Left handed (%)` = round(`Left-handed` * 100 / Total, 2)) %>%
  kable() %>%
  kable_styling(full_width = TRUE)
```

Table 2.4: **?(caption)**

(a)

group	Total	Left-handed	Left handed (%)
control	40	4	10
patient	30	9	30

Table 2.5: **?(caption)**

(a)

	L	R	Sum
control	4	36	40
patient	9	21	30
Sum	13	57	70

2.4 Contingency table

Shows the multivariate frequency distribution of variables

```
# use table() function to create contingency table
table.con <- table(data.handedness$group, data.handedness$handedness)
table.con <- addmargins(table.con)
table.con %>% kable() %>%
  kable_styling(full_width = TRUE)
```

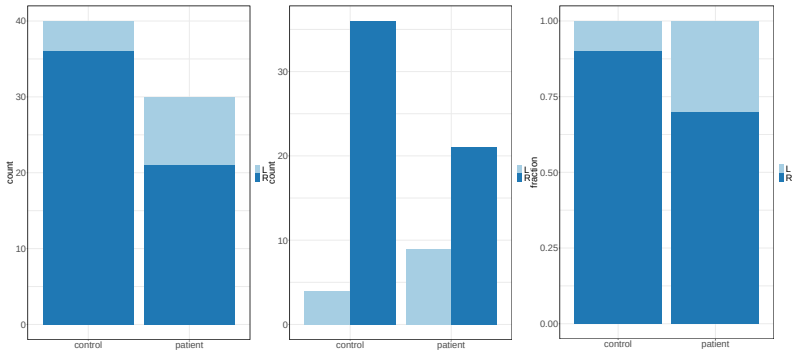
2.5 Bar chart with 2 categorical variables

Again, we can visualize the frequencies using bar charts.

```
data.handedness %>%
  ggplot(aes(x=group, fill=handedness)) +
  geom_bar() +
  theme_bw() +
  xlab("") +
  scale_fill_brewer(palette = "Paired") +
  my.ggtheme
```

```r
# another way of using bar charts: side by side bars
data.handedness %>%
  ggplot(aes(x=group, fill=handedness)) +
  geom_bar(position = "dodge") +
  theme_bw() +
  xlab("") +
  scale_fill_brewer(palette = "Paired") +
  my.ggtheme

# another way of using bar charts: showing fractions instead of counts
data.handedness %>%
  ggplot(aes(x=group, fill=handedness)) +
  geom_bar(position = "fill") +
  theme_bw() +
  xlab("") +
  ylab("fraction") +
  scale_fill_brewer(palette = "Paired") +
  my.ggtheme
```



(a) stacked bars (b) side-by-side bars (c) bars showing fractions instead of counts

Figure 2.2: Bar charts summarizing handedness variable in control and patients

2.6 Mosaic plot

Mosaic plots display contingency tables

```
# recreate contingency table to remove margins stats
table.con <- table(data.handedness$group, data.handedness$handedness)

# draw mosaic plot
font.scale <- 2
mosaicplot(table.con, col = "#a6cee3",
           main = "",
           cex.axis = font.scale)
```

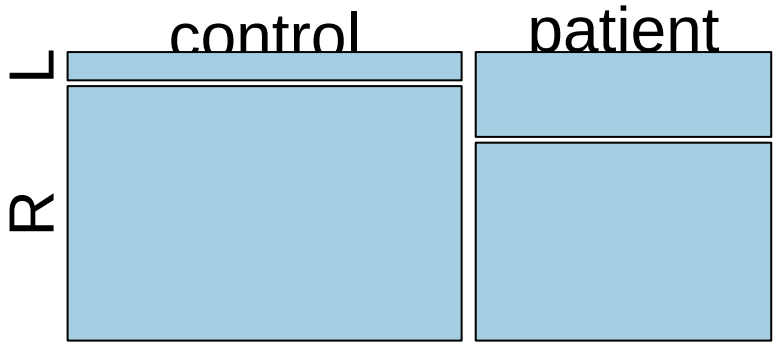
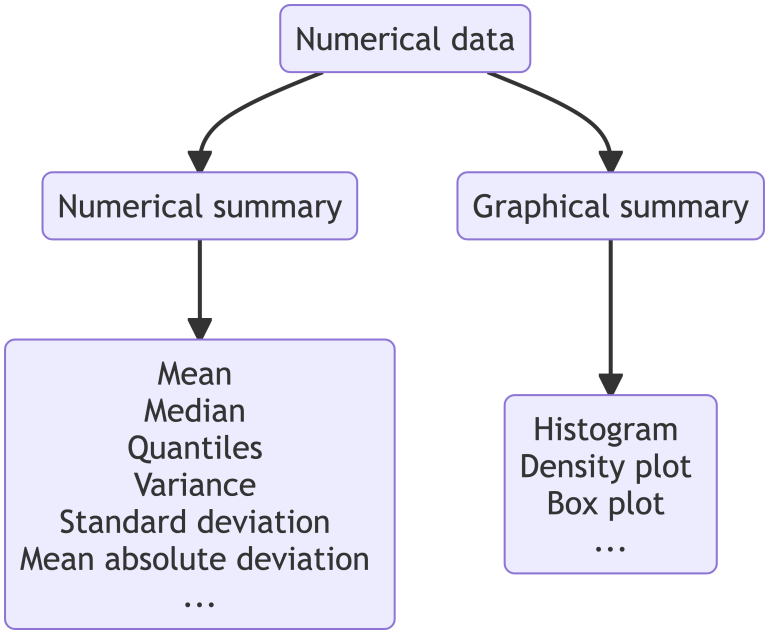


Figure 2.3: Mosaic plot showing contigency table

3 Numerical data

```
# load libraries
library(tidyverse)
library(kableExtra)
library(ggplot2)
library(ggbeeswarm)
library(gridExtra)
```

Numerical data, both discrete and continuous, can be visualized and summarized in many ways. Common plots include histograms, density plots, box plots and scatter plots. Summary statistics include mean, median, quantiles, variance, standard deviation and median absolute deviation.



Example 3.1 (Throwing 10 dice)**.** Let's imagine that we have 10 dice and we throw them all at once. We count and record the total number of dots and repeat the whole process 100 times.

The counts for the first few runs are:

```
# simulate throwing 10 dice, counting dots and repeating the experiment 100 times
set.seed(123)
sample.sum10dice <- replicate(100, sum(sample(1:6, 10, replace = TRUE)))
data.sum10dice <- data.frame(run = 1:length(sample.sum10dice), sumcounts = sample.sum10dice)

head(data.sum10dice)
```

```
  run sumcounts
1   1        40
2   2        29
3   3        33
4   4        27
5   5        40
6   6        27
```

3.1 Strip plot, Jittered strip plot & Beeswarm plot

When the data set is not very big, i.e. does not contain millions of measurements for a given numerical variable of interest, it can be useful to plot all measurements. This can be done in a **1D scatter plot**, called a strip plot or a dot plot.

```
# define generic ggplot theme
font.size <- 12
col.blue.light <- "#a6cee3"
col.blue.dark <- "#1f78b4"
my.ggtheme <- theme(axis.title = element_text(size = font.size),
        axis.text = element_text(size = font.size),
        legend.text = element_text(size = font.size),
        legend.title = element_blank(),
```

```
        legend.position = "top")

# plot strip plot
data.sum10dice %>%
  ggplot(aes(x = "", y = sumcounts)) +
  geom_point() +
  theme_bw() +
  ylab("dot counts (10 dice)") +
  xlab("") +
  my.ggtheme
```

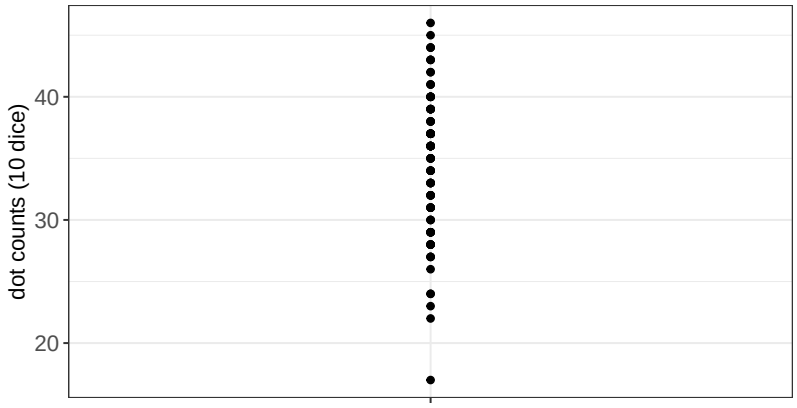


Figure 3.1: Strip plot showing sum of dots across 10 dice being thrown at once 100 times

As some measurements are repeated, e.g. we count 40 dots in our first attempt in our 40 attempt, the measurements on the strip plot are shown on top of each other and we cannot see them all. A **jittered strip plot** attempts to reduce overlays by randomly moving data points by small amounts to the left and right.

```
# plot jittered strip plot
data.sum10dice %>%
  ggplot(aes(x = "", y = sumcounts)) +
  geom_jitter(height = 0, width = 0.35) +
  theme_bw() +
  ylab("dot counts (10 dice)") +
  xlab("") +
  my.ggtheme
```

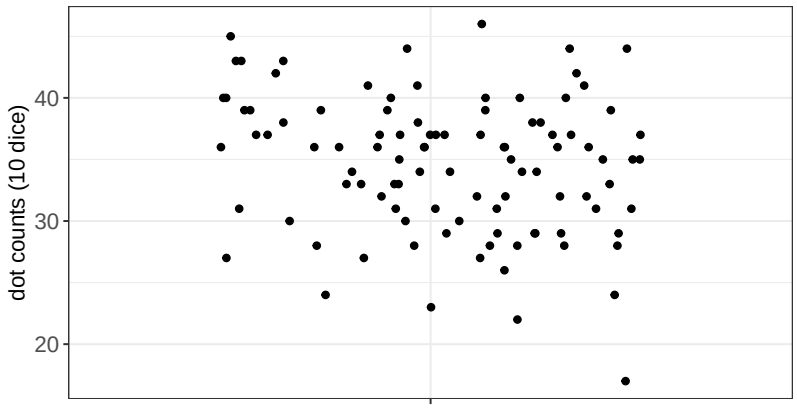


Figure 3.2: Jittered strip plot showing sum of dots across 10 dice being thrown at once 100 times

In a jittered strip plot some overlays may still occur, as the data points are moved at random. Further, many data points are moved unnecessarily. In a **beeswarm plot** data points are moved only when necessary, and even then the data point is only moved the minimum distance necessary to avoid overlays

```
# plot beeswarm
data.sum10dice %>%
  ggplot(aes(x = "", y = sumcounts)) +
  geom_beeswarm(cex = 2) +
  theme_bw() +
  ylab("dot counts (10 dice)") +
  xlab("") +
  my.ggtheme
```

Figure 3.3: Beeswarm showing sum of dots across 10 dice being thrown at once 100 times

3.2 Histogram & density plot

A **histogram** bins the data and counts the number of observations that fall into each bin.

```
# plot histogram
data.sum10dice %>%
  ggplot(aes(x = sumcounts)) +
  geom_histogram(binwidth = 5, center = 32.5, color = "white", fill = col.blue.dark) +
  theme_bw() +
  xlab("dot counts (10 dice)") +
  my.ggtheme

# alternatively un-comment to plot histogram with hist(), base R plot
# hist(data.sum10dice$counts, main = "", xlab="counts")
```

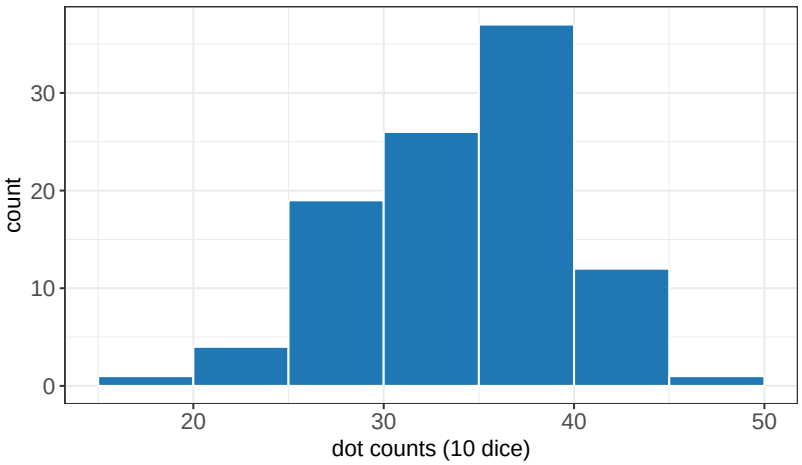


Figure 3.4: Histogram summarizing the sum of dots across 10 dice being thrown at once 100 times

A **density plot** is like a smoothed histogram where the total area under the curve is set to 1. A density plot is an approximation of a distribution.

```
# plot density plot
data.sum10dice %>% ggplot(aes(x = sumcounts)) +
  geom_density() +
  theme_bw() +
  xlab("dot counts (10 dice)") +
  my.ggtheme
```

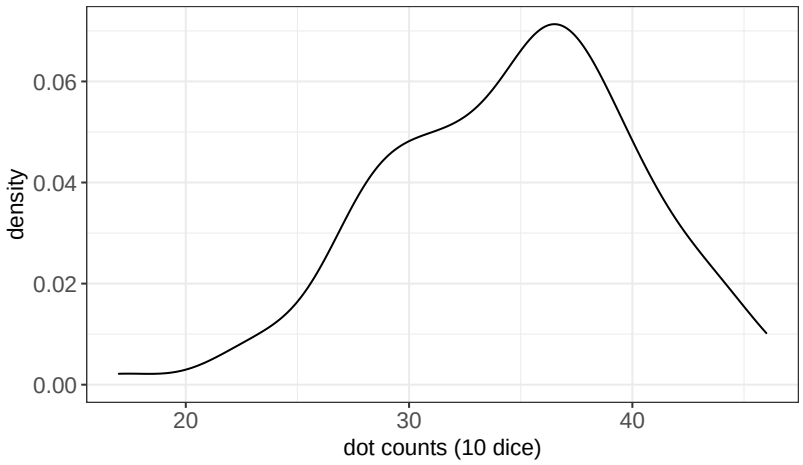


Figure 3.5: Density plot of dot counts across 10 dice

3.3 Box plot

A box plot, also called a box-and-whisker plot, shows a box covering 50% of the data and the center line is located at the **median**. The median value is a value such that 50% of the measurements are below the median.

The whiskers extend to the most extreme data point or at most 1.5 times the length of the box. (Note that 1.5 is the default in both ggplot and basic R graphics, but it is also a number that can be changed.) Any measurements further out are shown as outliers. A box plot is based on both measures of location and of spread (more about these in the following chapters).

Example 3.2 (Lab mice (cont.))**.** Let's go back to our lab mice example and focus on the weight measurements that we have observed for our male and female mice over a period of time. How can we summarize the data in a graphical way?

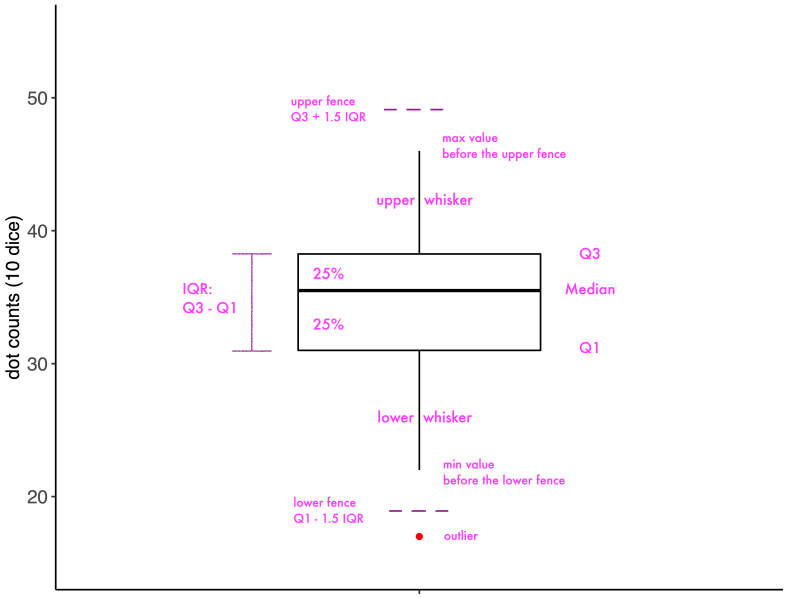


Figure 3.6: Box plot

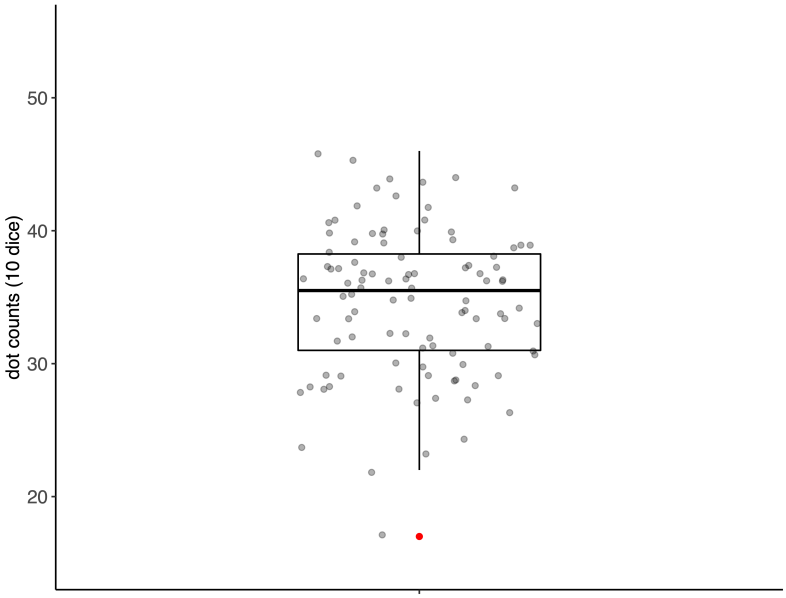


Figure 3.7: Box plot overlayed on a jittered strip plot

3.4 Strip plot, jittered stirp plot & beeswarm stratified by group

```r
# read in mice data
data.mice <- read_csv("data/mice.csv")
data.mice <- data.mice %>%
  mutate(weight = round(weight,2))

# select weights at week 5
data.mice.week5 <- data.mice %>%
  filter(week == 5)

# strip plot stratified by sex
p.striplot <- data.mice.week5 %>%
  ggplot(aes(x = sex, y = weight)) +
  geom_point() +
  theme_bw() +
  xlab("") +
  ylab("weigth (g)") +
  my.ggtheme

# jittered strip plot stratified by sex
p.jitter <- data.mice.week5 %>%
  ggplot(aes(x = sex, y = weight)) +
  geom_jitter(height = 0, width = 0.2) +
  theme_bw() +
  xlab("") +
  ylab("weigth (g)") +
  my.ggtheme

# beeswarm stratified by sex
p.beeswarm <- data.mice.week5 %>%
  ggplot(aes(x = sex, y = weight)) +
  geom_beeswarm(cex = 3) +
  theme_bw() +
  xlab("") +
  ylab("weigth (g)") +
  my.ggtheme
```

```
grid.arrange(arrangeGrob(p.striplot, p.jitter, p.beeswarm,ncol=3))
```

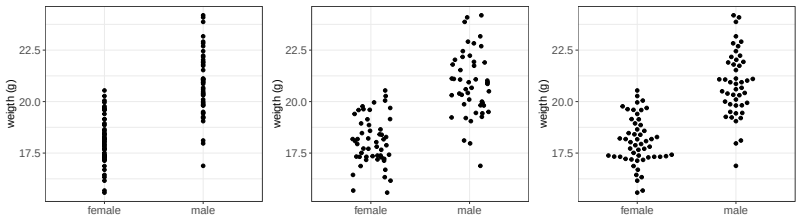


Figure 3.8: Strip plot, jittered strip plot and beeswarm plot showing mice weights at week 5 stratified by sex

3.5 Histogram & density plot stratified by group

```
# plot histogram
p.hist <- data.mice.week5 %>%
  ggplot(aes(x=weight, fill=sex)) +
  geom_histogram(bins=15, color="white", alpha = 0.6) +
  xlab("weight (g)") +
  theme_bw() +
  scale_fill_brewer(palette = "Dark2") +
  my.ggtheme


p.density <- data.mice.week5 %>%
  ggplot(aes(x=weight, fill=sex)) +
  geom_density(alpha = 0.6) +
  xlab("weight (g)") +
  theme_bw() +
  scale_fill_brewer(palette = "Dark2") +
  my.ggtheme

grid.arrange(arrangeGrob(p.hist, p.density, ncol=2))
```

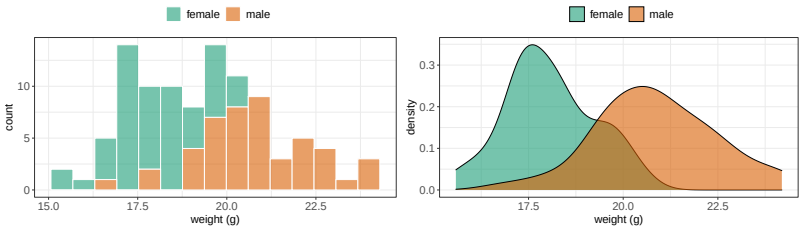


Figure 3.9: Histogram and density plot summarizing the weights of mice at week 5, stratified by sex

3.6 Box plot stratified by group

```
# plot box plot
data.mice.week5 %>%
  ggplot(aes(y=weight, fill=sex)) +
  geom_boxplot(alpha = 0.6) +
  xlab("weight (g)") +
  theme_bw() +
  scale_fill_brewer(palette = "Dark2") +
  my.ggtheme
```

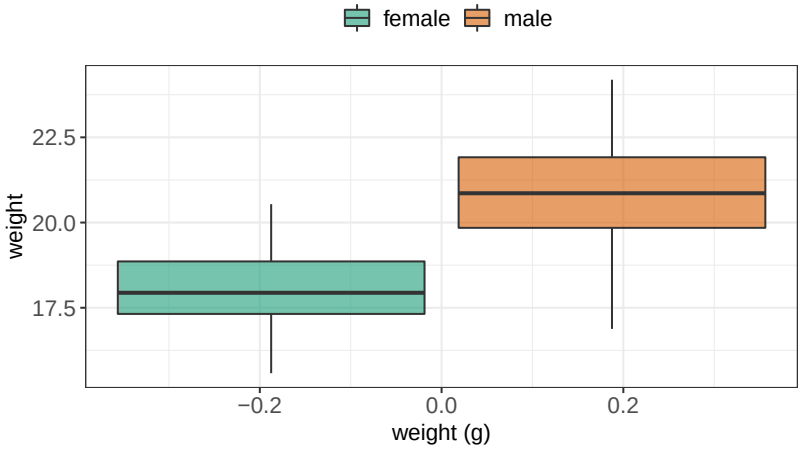


Figure 3.10: Box plot of mice weights at week 5 stratified by sex

3.7 Scatter plot: 2 numerical variables

Scatter plots are useful when studying a relationship (association) between two numerical variables. Let's add some data on our mice length and have a look at the relationship between mice weight and length at week 10.

```r
# simulate mice length data (based on normal distribution)
data.mice.week10 <- data.mice %>%
  filter(week == 10) %>%
  mutate(length = 7.3+weight/20+rnorm(100,0,0.1))

# plot scatter plot
data.mice.week10 %>%
  ggplot(aes(x = weight, y = length)) +
  geom_point() +
  xlab("weigth (g)") +
  ylab("length (cm)") +
  theme_bw() +
  my.ggtheme
```

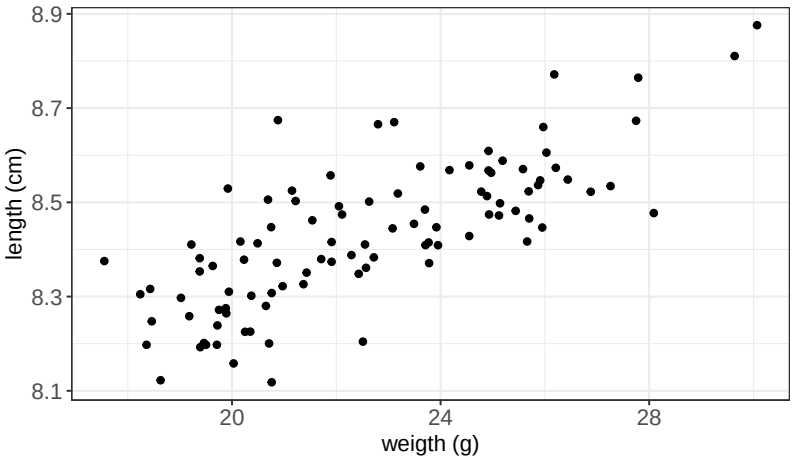


Figure 3.11: Scatter plot showing a relationship between mice weight and length at week 10

```r
# plot scatter plot
data.mice.week10 %>%
  ggplot(aes(x = weight, y = length, color = sex)) +
  geom_point() +
  xlab("weigth (g)") +
  ylab("length (cm)") +
  theme_bw() +
  scale_color_brewer(palette = "Dark2") +
  my.ggtheme
```

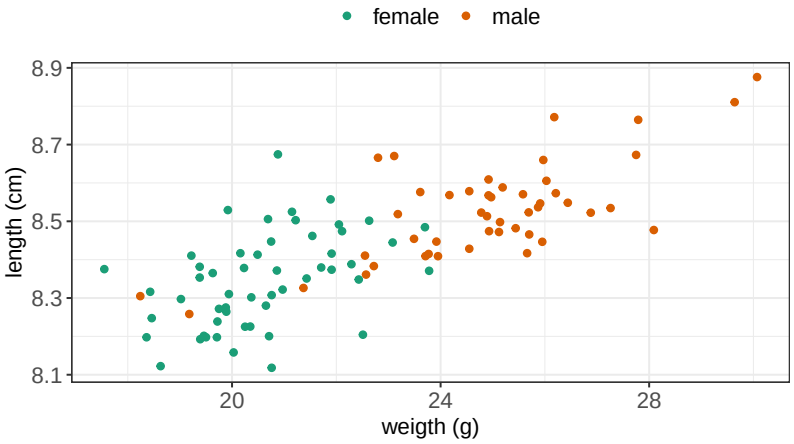


Figure 3.12: Scatter plot showing a relationship between mice weight and length at week 10 stratified by sex

Sometimes, it is useful to connect the observations in the order in which they appear, e.g. when analyzing time series data.

```
# select four mice, with ids 17, 18 and 19
data.mice.4 <- data.mice %>%
  subset(id %in% 16:19)

# plot a line plot
data.mice.4 %>%
  ggplot(aes(x=week, y=weight, group=id)) +
  geom_point() +
  geom_line() +
  ylab("weigth (g)") +
  theme_bw() +
  my.ggtheme
```

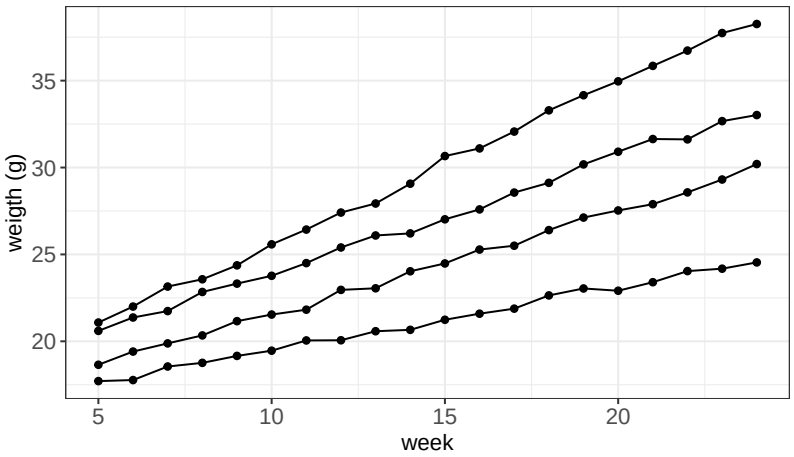


Figure 3.13: Line plot over age of mouse vs. weight (g) for four mice in the experiment

```
# plot a line plot
data.mice.4 %>%
  ggplot(aes(x=week, y=weight, group=id, color = sex)) +
  geom_point() +
  geom_line() +
  ylab("weigth (g)") +
  theme_bw() +
  scale_color_brewer(palette = "Dark2") +
  my.ggtheme
```

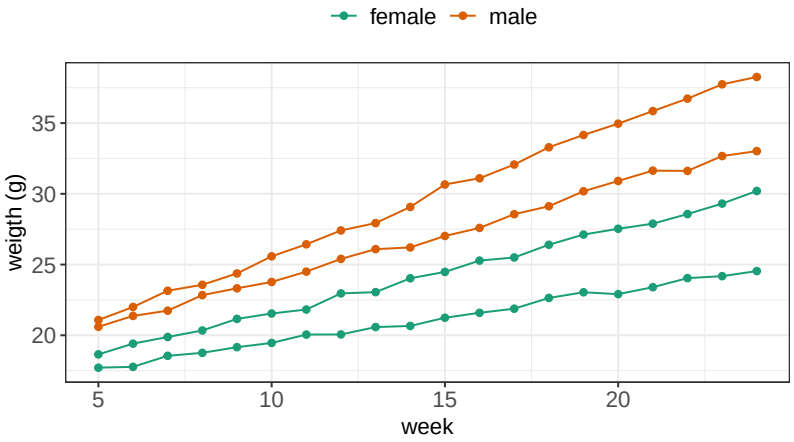


Figure 3.14: Line plot over age of mouse vs. weight (g) for four mice in the experiment

4 Measures of location

```
# load libraries
library(tidyverse)
library(magrittr)
library(kableExtra)
library(ggplot2)
library(ggbeeswarm)
library(gridExtra)
library(reshape2)
```

It is not always easy to get a "feeling" for a set of numerical measurements unless we summarize the data in a meaningful way. Diagrams, as shown in the previous chapter, are often a good starting point. We can further condense the information by reporting what constitutes a representative value. If we also know how widely scattered the observations are around it, we can formulate an image of data. The **average** is a general term for a measure of **location** and some common ways of calculating the average are mode, mean and median.

4.1 Mode

Mode values is the value that most common occurs across the measurements. It can be found for numerical and categorical data types.

4.2 Median

Median value divides the ordered data values into two equally sized groups, so 50% of the values are below and 50% are above the median value.

4.3 Arthimetic mean & weighted mean

The arithmetic mean, also commonly referred to as to mean, is calculated by adding up all the values and diving this by the number of values in the data set.

Mathematically, for n observations $x_1, x_2, \ldots, x_n$, the arithmetic mean value is calculated as:

$$\bar{x} = \frac{x_1 + x_2 + \cdots + x_n}{n} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.1}$$

As all the values equally contribute to the calculations, the arithmetic mean value is easily affected by outliers and is distorted by skewed distributions. Sometimes, the **weighted mean** may be more useful, as it allows weights to certain values of the variable of interest. We attach a weight, w_i to each of the observed values, x_i, in our sample, to reflect this importance and define the weighted mean as:

$$\bar{x} = \frac{w_1 x_1 + w_2 x_2 + \ldots + w_n x_n}{w_1 + w_2 + \ldots + w_n} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} \tag{4.2}$$

Example 4.1 (Lab mice (cont.))**.** Let's revisit our lab mice example and focus on data from week 5. What's the mode value of sex variable? What is median value of the first 9 measures of mice weight at week 5? How about for the first 10 measurements? And what is the average weight of mice during that week?

```
# read in mice data
data.mice <- read_csv("data/mice.csv")
data.mice <- data.mice %>%
  mutate(weight = round(weight,2))

# narrow data set to week 5 measurements
```

```
data.mice.week5 <- data.mice %>%
  filter(week == 5)
```

Mode value

We can find mode value by counting how many times we observe males and females among our mice. The mode value is the most commonly occurring one, here "females". We have see from counting the values, that we have 53 female and 47 male mice.

```
# find mode value
data.mice.week5 %>%
  group_by(sex) %>%
  tally() %>%
  arrange(desc(n)) %>%
  print()
```

```
# A tibble: 2 x 2
  sex        n
  <chr>  <int>
1 female    53
2 male      47
```

Median value

We can find median value by:

- ordering values
- and finding the middle value, if the number of measurement is odd
- and taking the arithmetic average of the two middle measurements, if the number of measurements is even

Median value for the first 9 measurements of mice weight at week 5:

```
# find median value for the first 9 measurements (odd)
data.mice.week5 %>%
  slice(1:9) %>% # select first 9 measurements
  arrange(weight) %>% # arrange by weight
```

```
  slice(5) %>%  # get the middle, fifth measurement
  pull(weight) %>%
  print() # print weight value for the 5th measurement (median)
```

```
[1] 20.33
```

```
# # alternatively use median() function
# data.mice.week5 %>%
#   slice(1:9) %>% # select first 9 measurements
#   arrange(weight) %>% # arrange by weight
#   pull(weight) %>%
#   median()
```

Median value for the first 10 measurements of mice weight at week 5:

```
# find median value for the first 9 measurements (odd)
data.mice.week5 %>%
  slice(1:10) %>% # select first 9 measurements
  arrange(weight) %>% # arrange by weight
  slice(5,6) %$%  # get 2 middle values, fifth & sixth
  mean(weight) %>%
  print() # print weight value for the 5th measurement (median)
```

```
[1] 19.69
```

Arithmetic mean and weighted mean

To calculate the arithmetic mean we can follow Equation 4.1

```
# calculate arithmetic mean following equation
x <- data.mice.week5 %>%
  pull(weight) # extract weight observations
n <- length(x) # number of observations
x.bar <- sum(x) / n # calculate mean
print(x.bar)
```

```
[1] 19.3752
```

or use basic `mean()` function in R:

```
# or alternatively use mean() function
# calculate arithmetic mean
data.mice.week5 %$%
  mean(weight) %>%
  print()
```

```
[1] 19.3752
```

The above arithmetic mean value may be however not best to reflect an average mice weight since we do not have equal numbers of males and females in the study. Here we know,or rather assume for the purpose of this example, that it is equally likely to find male and females in mice population and hence, in our experiment, males are underrepresented.

We can calculate weighted mean to account for group sizes. We assign weights so that they sum up to 100. The males and females group should have equal influence, so 50/50. As we have 53 females, the female weight is $w_f = 50/53 = 0.9433962$ and male weight is $w_m = 50/47 = 1.06383$. The weighted mean can now be calculated following Equation 4.2 and is equal to:

```
# number of females
n.females <- data.mice.week5 %>%
  filter(sex == "female") %>%
  nrow()

# number of males
n.males <- data.mice.week5 %>%
  filter(sex == "male") %>%
  nrow()

# add weights to observations
data.mice.week5.addweight <- data.mice.week5 %>%
  mutate(w = ifelse(sex == "male", 50/n.males, 50/n.females)) %>% # assign weights
  mutate(wx = weight * w) %$% # multiply weight by their weights values
```

```
    mean(wx) %>% # average the weighted measurements
    print()
```

```
[1] 19.45934
```

> **Warning**
>
> Note that several very different distributions can still have the same mean value.

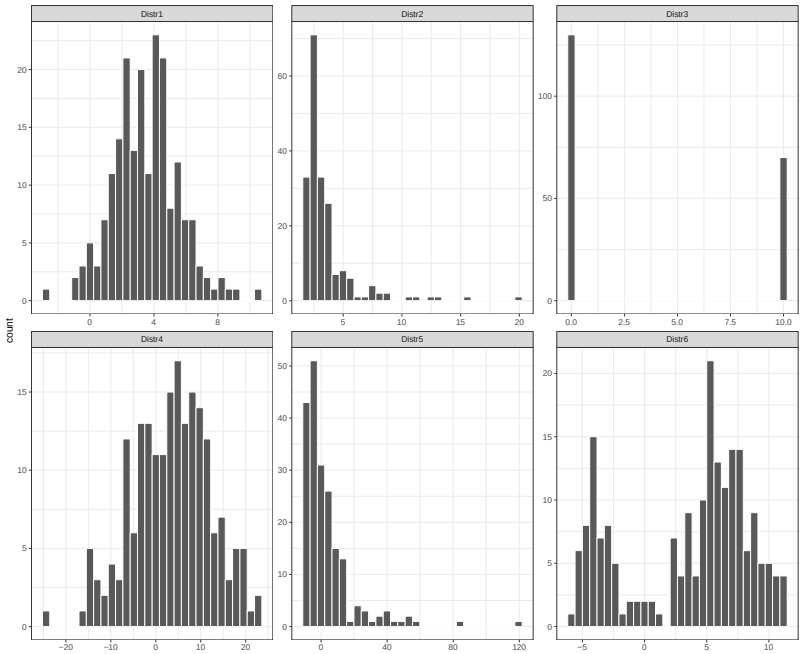


Figure 4.1: Examples of various distributions having the same mean value of 3.5

5 Measures of spread

```
# load libraries
library(tidyverse)
library(magrittr)
library(kableExtra)
library(ggplot2)
library(gridExtra)
```

5.1 Quartiles

Quartiles are the three values that divide the data values into four equally sized groups.

- **Q1**. Lower quartile. 25% of values are below Q1. Divides the values below the median into equally sized groups.
- **Q2**. Median. 50% of values are below Q2 and 50% are above Q2. Q2 is the median that we have seen before.
- **Q3**. Upper quartile. 75% of values are below Q3. Divides the values above the median into equally sized groups.
- **IQR**. Interquartile range, midspread, middle 50%. IQR = Q3 - Q1.

Example 5.1 (Lab mice (cont.))**.** Going back to the lab mice example. What are the three quartiles of mice weight at week 5?

```
# read in mice data
data.mice <- read_csv("data/mice.csv")
data.mice <- data.mice %>%
```

```
  mutate(weight = round(weight,2))

# narrow data set to week 5 measurements
data.mice.week5 <- data.mice %>%
  filter(week == 5)

# calculate quartiles
data.mice.week5 %>%
  summarise(x = quantile(weight, c(0.25, 0.5, 0.75))) %>%
  print()
```

```
# A tibble: 3 x 1
      x
  <dbl>
1  17.8
2  19.3
3  20.6
```

We can check if the values agree with the box plot.

```
# define generic ggplot theme
font.size <- 12
col.blue.light <- "#a6cee3"
col.blue.dark <- "#1f78b4"
my.ggtheme <- theme(axis.title = element_text(size = font.size),
        axis.text = element_text(size = font.size),
        legend.text = element_text(size = font.size),
        legend.title = element_blank(),
        legend.position = "top")

data.mice.week5 %>%
  ggplot(aes(x = "", y = weight)) +
  geom_boxplot() +
  xlab("") +
  ylab("weight (g)") +
  theme_bw() +
  my.ggtheme
```

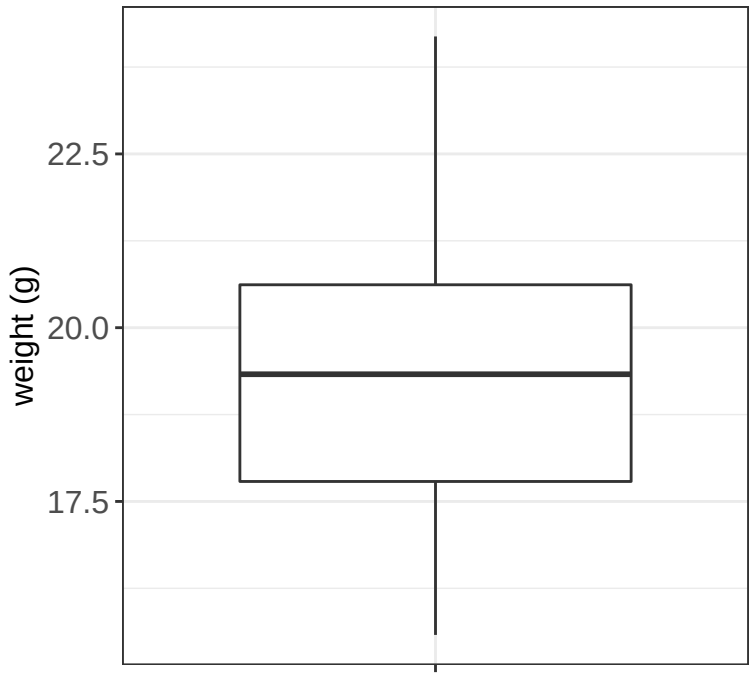


Figure 5.1: Bar plot of mice weights at week 5

5.2 Variance and standard deviation

The **variance** of a set of observations is their mean squared distance from the mean value:

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2. \tag{5.1}$$

```
data.xy <- data.mice.week5 %>%
  slice(1:10) %>%
  rename(y = weight) %>%
  rename(x = id)

y.bar <- mean(data.xy$y)

data.xy %>%
ggplot(aes(x=x, y=y)) +
  geom_segment( aes(x=x, xend=x, y=y, yend=y.bar), color="grey") +
```

```
  geom_point(color=col.blue.dark, size=4) +
  geom_hline(yintercept=y.bar) +
  theme_bw() +
  theme(
    panel.grid.major.x = element_blank(),
    panel.border = element_blank(),
    axis.ticks = element_blank(),
    axis.text.y = element_blank()) +
  xlab("") +
  ylab("Length (cm)") +
  coord_flip() +
  my.ggtheme
```

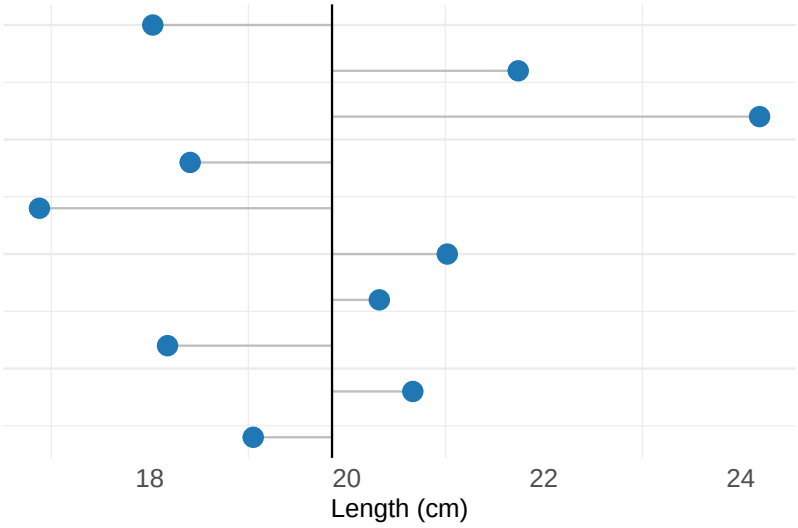


Figure 5.2: First ten measurments of mice weight at week 5. Grey lines show the distance to the mean value.

The variance is measured in the square of the unit in which x was measured. Another common measure using the same unit as x is **standard deviation**, defined as the square root of the variance:

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{5.2}$$

Typically, we regard the collection of observations $x_1, \ldots, x_n$ as a **sample** drawn from a large **population** of possible observations. It has been shown theoretically that we obtain a better

sample estimate of the **population standard deviation** if we divide by $(n-1)$. So the denominator n is commonly replaced by $n-1$ and the **sample standard deviation** is calculated instead:

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}. \tag{5.3}$$

To reiterate:

- we calculate standard deviation, σ, when we consider our observations to be entire population (Equation 5.2)
- we calculate sample standard deviation, s, when we consider our observations a random sample from a larger population, as a better estimate of the standard deviation of the larger population (Equation 5.3)

Example 5.2 (Lab mice (cont.))**.** Let's calculate variance, standard deviation and sample standard deviation for mice weight at week 5.

```
# get weights measurements at week 5
weight.week5 <- data.mice.week5 %>%
  pull(weight)

# number of observations
n <- length(weight.week5)

# calculate mean (arithmetic)
weight.mean <- mean(weight.week5)

# calculate variance following variance equation
sigma2 <- (sum((weight.week5 - weight.mean)^2))/(n)

# calculate standard deviation following standard deviation equation
sigma <- sqrt((sum((weight.week5 - weight.mean)^2))/(n))

# calculate sample standard deviation following sample standard deviation equation
```

```r
s <- sqrt((sum((weight.week5 - weight.mean)^2))/(n-1))
# sd(weight.week5) # we can alternatively use sd() function

# collect results
v.name <- c("sigma2", "sigma", "s")
v.values <- c(sigma2, sigma, s)
results <- data.frame(stats = v.name, value = v.values)
print(results)
```

```
   stats    value
1 sigma2 3.817581
2  sigma 1.953863
3      s 1.963706
```

6 Exercises

```
# load libraries
library(tidyverse)
library(kableExtra)
library(ggplot2)
library(ggbeeswarm)
library(gridExtra)
```

Exercise 6.1 (Babies)**.** Consider the below data and summarize each of the variables. Note, there is no need to use R here, just use pen and paper, maybe use R as a calculator.

```
baby %>% kable() %>%
  kable_styling("striped", full_width = FALSE)
```

id	smoker	baby weight (kg)	gender	mother weight (kg)	mother age	parity	married
1	yes	2.8	F	64	21	2	yes
2	yes	3.2	M	65	27	1	yes
3	yes	3.5	F	60	31	2	yes
4	yes	2.7	F	73	32	0	yes
5	yes	3.3	M	59	39	3	yes
6	no	3.7	F	62	26	0	no
7	no	3.3	F	52	27	2	no
8	no	4.3	F	59	21	0	no
9	no	3.2	M	65	28	1	no
10	no	3.0	M	81	33	4	yes

Exercise 6.2 (Active substance)**.** The amount of active substance in a pill is stated by the manufacturer to be normally distributed with mean 12 mg and standard deviation 0.5 mg. You take a sample of five pills and measure the amount of active substance to be: 13.0, 12.3, 12.6, 12.5, 12.7 mg.

a) Compute the sample mean
b) Compute the sample variance
c) Compute the sample standard deviation

Exercise 6.3 (Mice)**.** a) Download the mice.csv data set and take a first look at the data. How large is the data.frame, how many rows/columns? What are the column names and what is the data type of each column? How many mice are described in the data set? Useful commands in R include `summary, View, dim, nrow, ncol, colnames`

b) The id column has identifiers for the mice and each mouse is described by many data points. Select a particular week, create a new data.frame with only weights of mice at this particular week. Plot the distribution of weights in at least one way. Useful commands in R include `subset, hist, density`

c) Summarize the entire data set using box plots

d) Can you think of another way to visualize the data set?

Exercise 6.4 (Primary Biliary Cholangitis (PBC))**.** Load the dataset pbc, from the survival() package. To read more about the data set you can read the help text.

```
library(survival)
data(pbc)
?pbc
```

a) Take a first look at the data, e.g. using `summary`. All variables, excep sex, are coded numerically. Is this correct? If not, which variables are really categorical? Use `factor` to change them into categorical variables. Run `summary` on the dataset again.
b) There are packages available for easily summarizing a data set, one of them is `tableone`. Install the package, load it and apply it to the pbc data set. Hint: the main function is called `CreateTableOne`
c) Plot the copper and bili values, both separately using an appropriate plot and together in a scatter plot.

Solutions: Descriptive statistics

Solution. Exercise 6.1

- Smokers: 5 (50%) yes
- baby weight (kg) mean (sd): 3.3 (0.44)
- gender: 6 (60%) F
- mother weight (kg) mean(sd): 64 (8.5)
- mother age mean(sd): 28.5 (5.8)
- partity mean(sd): 1.5 (1.4) could also be handled as categorical (ordinal) and report frequencies and percentages.
- married: 4 (40%) yes

Did you compute standard deviations that were slightly different? Then you probably computed the sample standard deviation, which could actually be what you want to report. When do you want to compute sample standard deviation?

Solution. Exercise 6.1

a. 12.62
b. 0.067
c. 0.26
d. 0.22
e. 0.0028