Intro and descriptive statistics

The ASTA team

Contents

1 Software

1.1 Rstudio

- Make a folder on your computer where you want to keep files to use in **Rstudio**. **Do NOT use Danish characters æ, ø, å** in the folder name (or anywhere in the path to the folder).
- Set the working directory to this folder: `Session -> Set Working Directory -> Choose Directory` (shortcut: Ctrl+Shift+H).
- Make the change permanent by setting the default directory in: `Tools -> Global Options -> Choose Directory`.

1.2 R extensions

- The functionality of **R** can be extended through libraries or packages (much like plugins in browsers etc.). Some are installed by default in **R** and you just need to load them.
- To install a new package in **Rstudio** use the menu: `Tools -> Install Packages`
- You need to know the name of the package you want to install. You can also do it through a command:

```
install.packages("mosaic")
```

- When it is installed you can load it through the `library` command:

```
library(mosaic)
```

- This loads the `mosaic` package which has a lot of convenient functions for this course (we will get back to that later). It also prints a lot of info about functions that have been changed by the `mosaic` package, but you can safely ignore that.

1.3 R help

- You get help via `?<command>`:

```
?sum
```

- Use `tab` to make **Rstudio** guess what you have started typing.
- Search for help:

```
help.search("plot")
```

- You can find a cheat sheet with the **R** functions we use for this course here.

2 Data

2.1 Data example

We use data about pengiuns from the R package palmerpenguins

```
pingviner <- palmerpenguins::penguins
pingviner
```

```
## # A tibble: 344 x 8
##    species island    bill_length_mm bill_depth_mm flipp~ body~ sex    year
##    <fctr>  <fctr>             <dbl>         <dbl>  <int> <int> <fct> <int>
##  1 Adelie  Torgersen           39.1          18.7    181  3750 male   2007
##  2 Adelie  Torgersen           39.5          17.4    186  3800 fema~  2007
##  3 Adelie  Torgersen           40.3          18.0    195  3250 fema~  2007
##  4 Adelie  Torgersen           NA            NA       NA    NA <NA>   2007
##  5 Adelie  Torgersen           36.7          19.3    193  3450 fema~  2007
##  6 Adelie  Torgersen           39.3          20.6    190  3650 male   2007
##  7 Adelie  Torgersen           38.9          17.8    181  3625 fema~  2007
##  8 Adelie  Torgersen           39.2          19.6    195  4675 male   2007
##  9 Adelie  Torgersen           34.1          18.1    193  3475 <NA>   2007
## 10 Adelie  Torgersen           42.0          20.2    190  4250 <NA>   2007
## # ... with 334 more rows
```

- What is fundamentally different about the the variables (columns) `species` and `body_mass_g`?

2.2 Data types

2.2.1 Quantitative variables

- The measurements have numerical values.
- Quantative data often comes about in one of the following ways:
  - **Continuous variables**: measurements of time, length, size, age, mass, etc.
  - **Discrete variables**: counts of e.g. words in a text, hits on a webpage, number of arrivals to a queue in one hour, etc.
- Measurements like this have a well-defined scale and in **R** they are stored as the type **numeric**.
- It is important to be able to distinguish between discrete count variables and continuous variables, since this often determines how we describe the uncertainty of a measurement.
- Are any of the measurements in our data set quantitative?

2.2.2 Categorical/qualitative variables

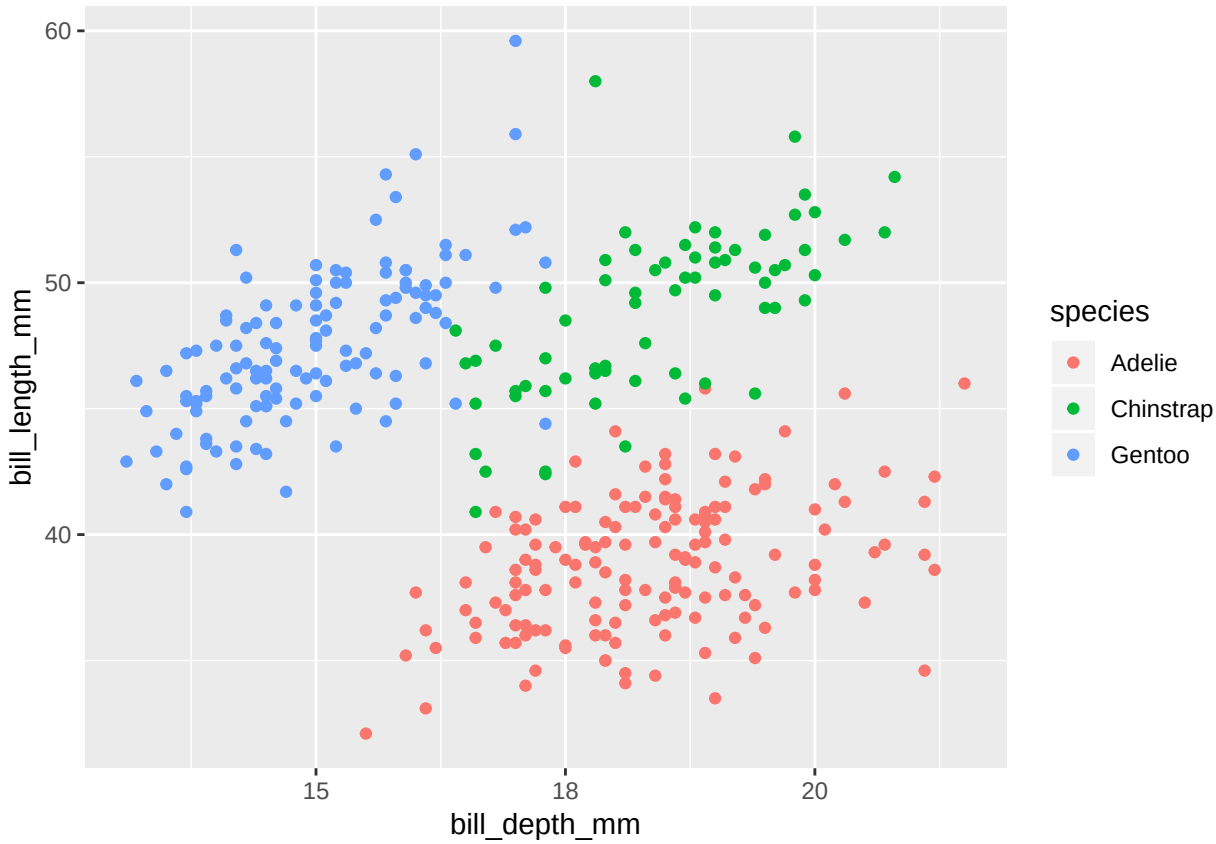
- The measurement is one of a set of given categories, e.g. sex (male/female), social status, satisfaction score (low/medium/high), etc.
- The measurement is usually stored (which is also recommended) as a **factor** in **R**. The possible categories are called **levels**. Example: the levels of the factor "sex" is male/female.
- Factors have two so-called scales:
  - **Nominal scale**: There is no natural ordering of the factor levels, e.g. sex and hair color.
  - **Ordinal scale**: There is a natural ordering of the factor levels, e.g. social status and satisfaction score. A factor in **R** can have a so-called **attribute** assigned, which tells if it is ordinal.
- Are any of the measurements in our data set categorical/qualitative?

3 Graphics for quantitative variables

3.1 Scatterplot

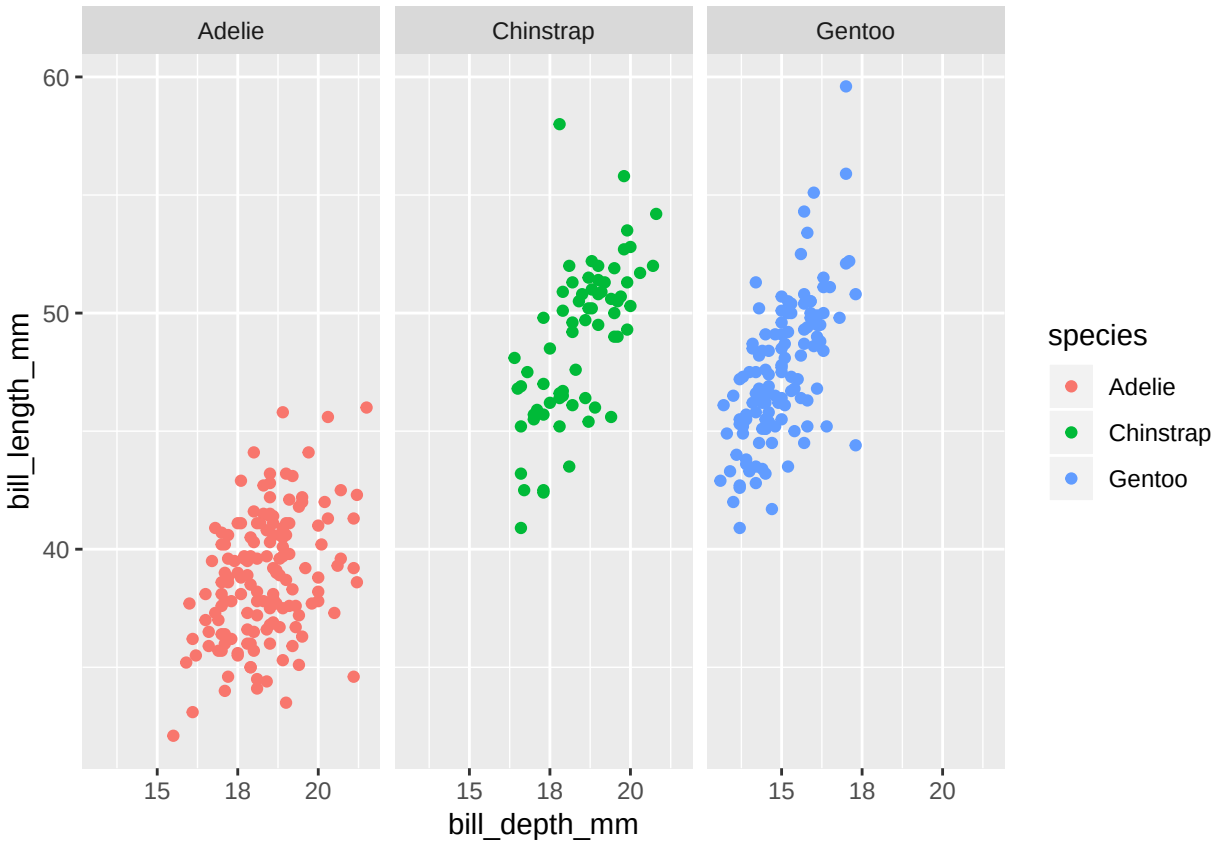
- To study the relation between two quantitative variables a scatterplot is used:

```
gf_point(bill_length_mm ~ bill_depth_mm, color = ~ species, data = pingviner)
```



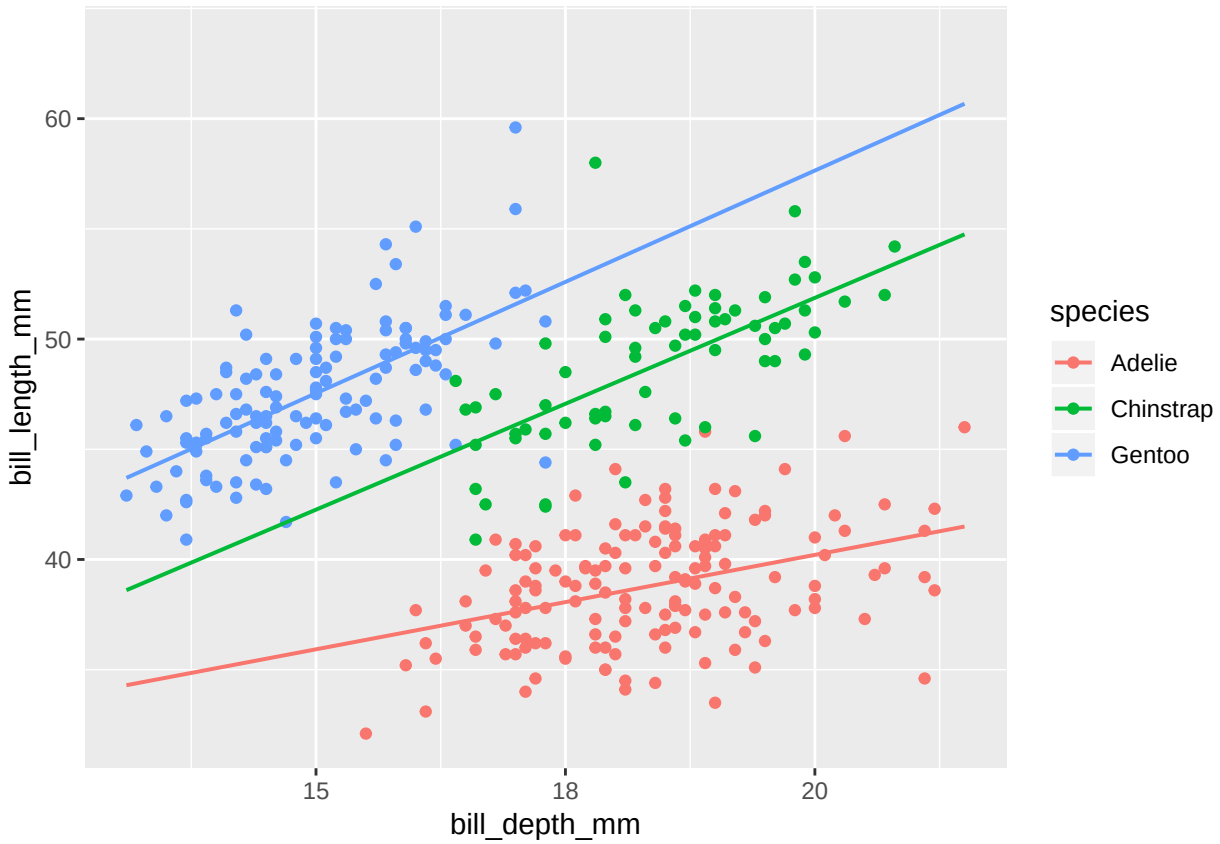
- We could also draw the graph for each species:

```
gf_point(bill_length_mm ~ bill_depth_mm | species, color = ~ species, data = pingviner)
```



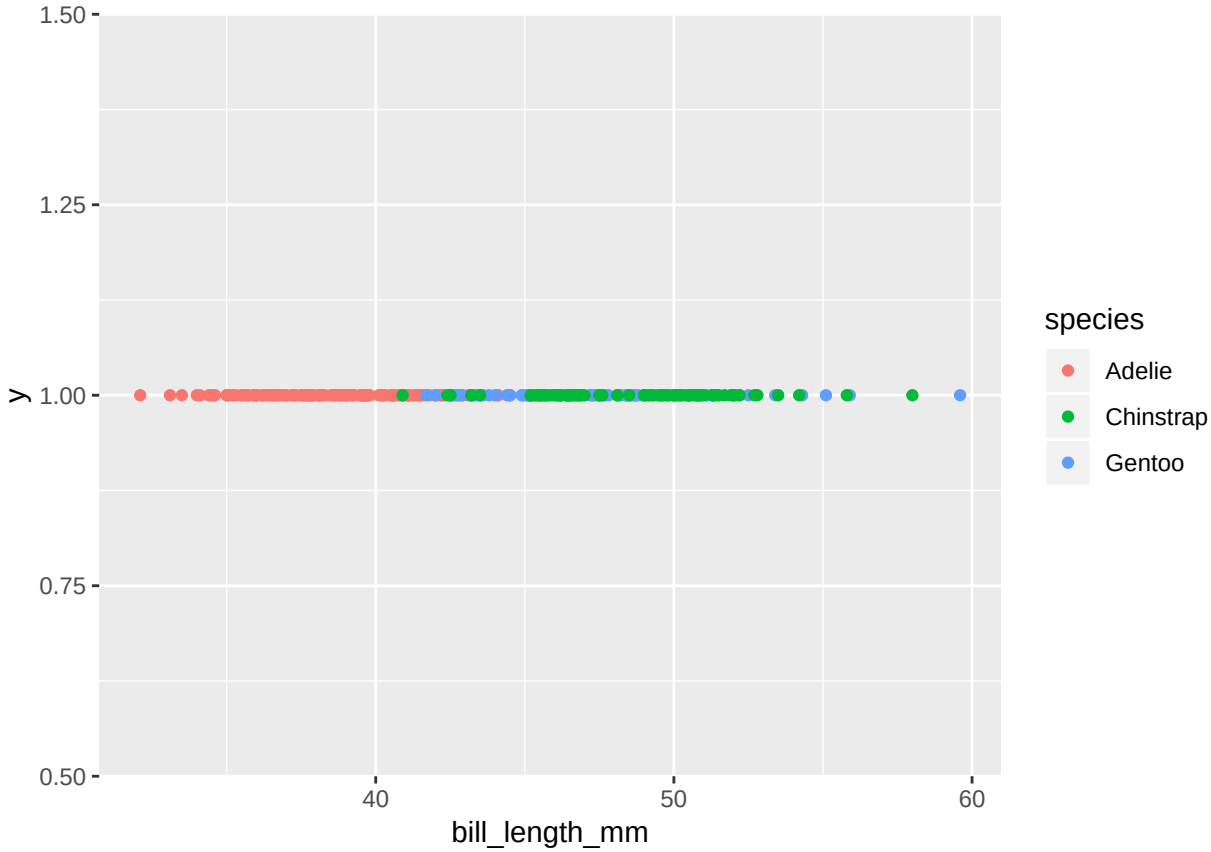
- If we want a regression line along with the points we can do:

```
gf_point(bill_length_mm ~ bill_depth_mm, color = ~ species, data = pingviner) %>%
  gf_lm()
```

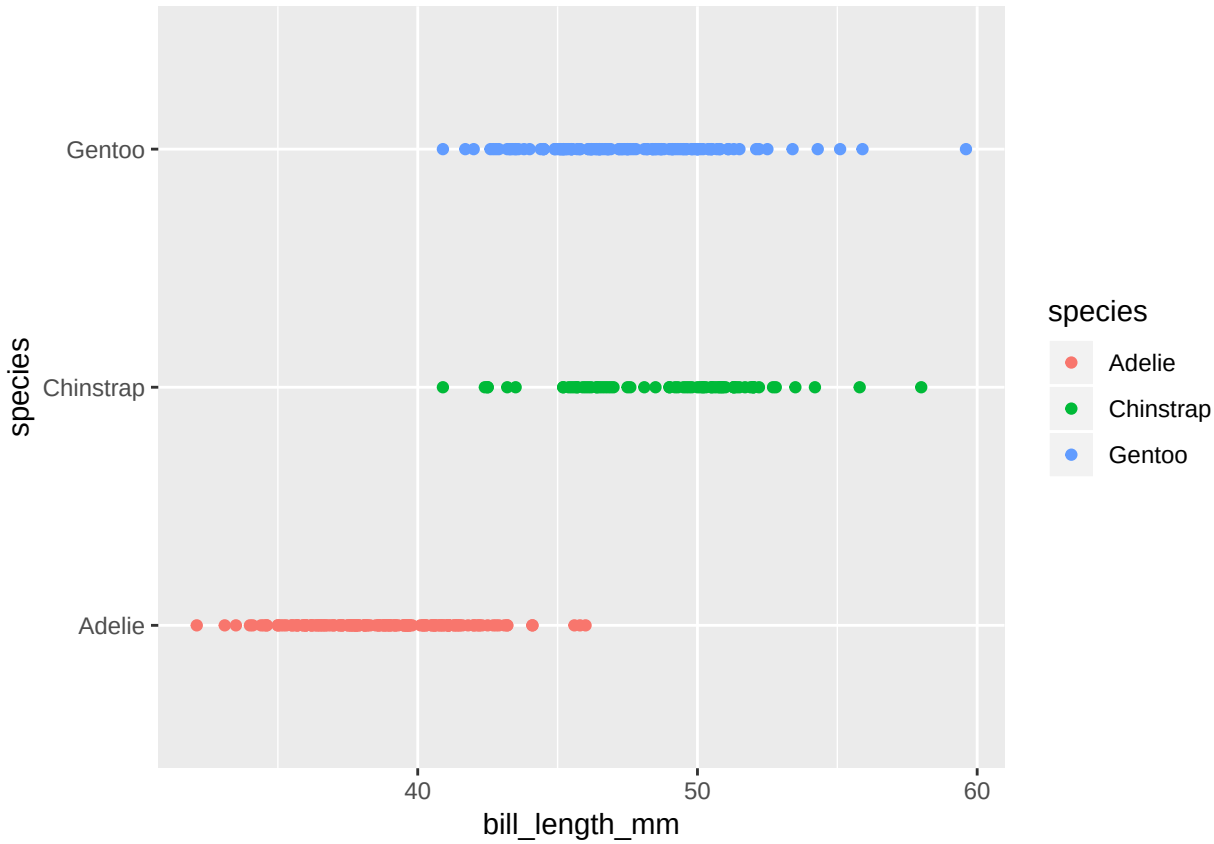


- A scatterplot is not very good for a single quantitative variable:

```
gf_point(1 ~ bill_length_mm, color = ~ species, data = pingviner)
```



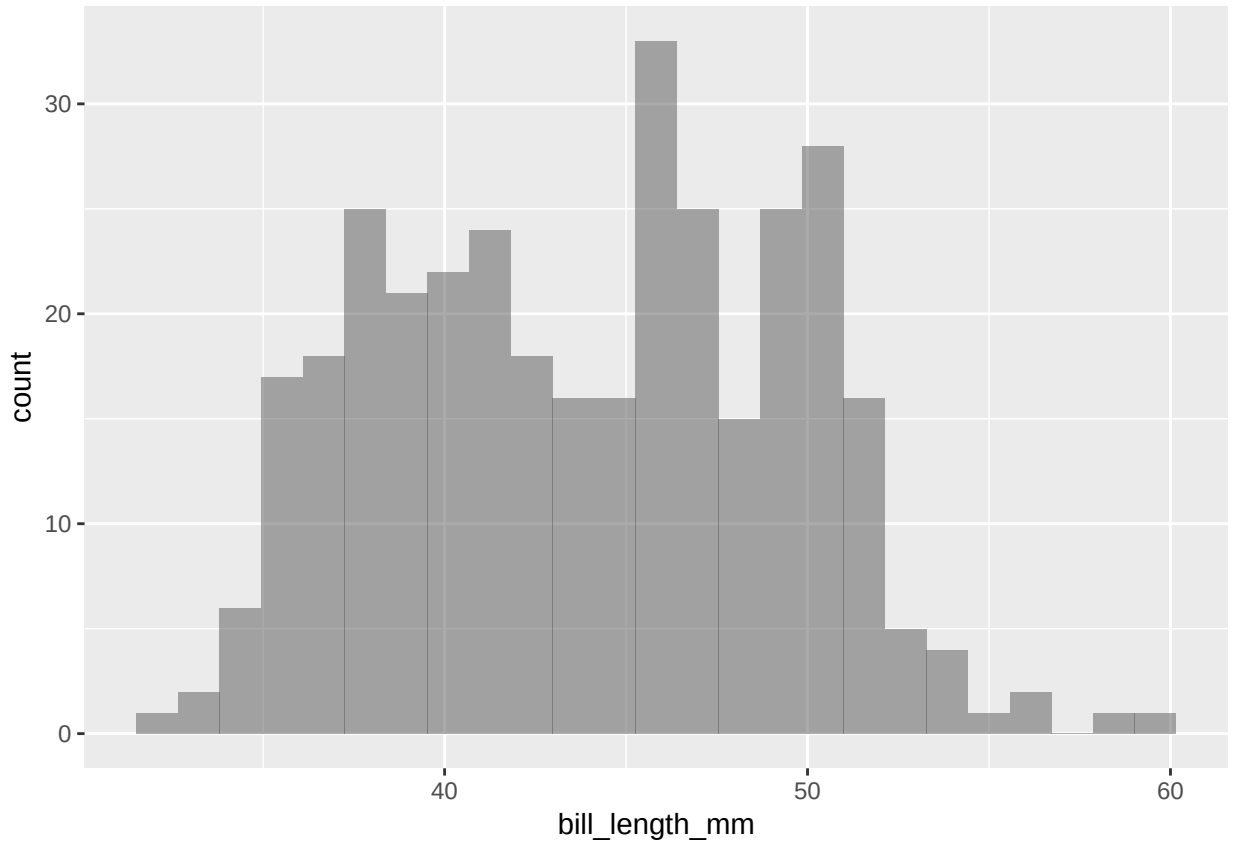

```
gf_point(species ~ bill_length_mm, color = ~ species, data = pingviner)
```



3.2 Histogram

- For a single quantitative variable a histogram offers more details:

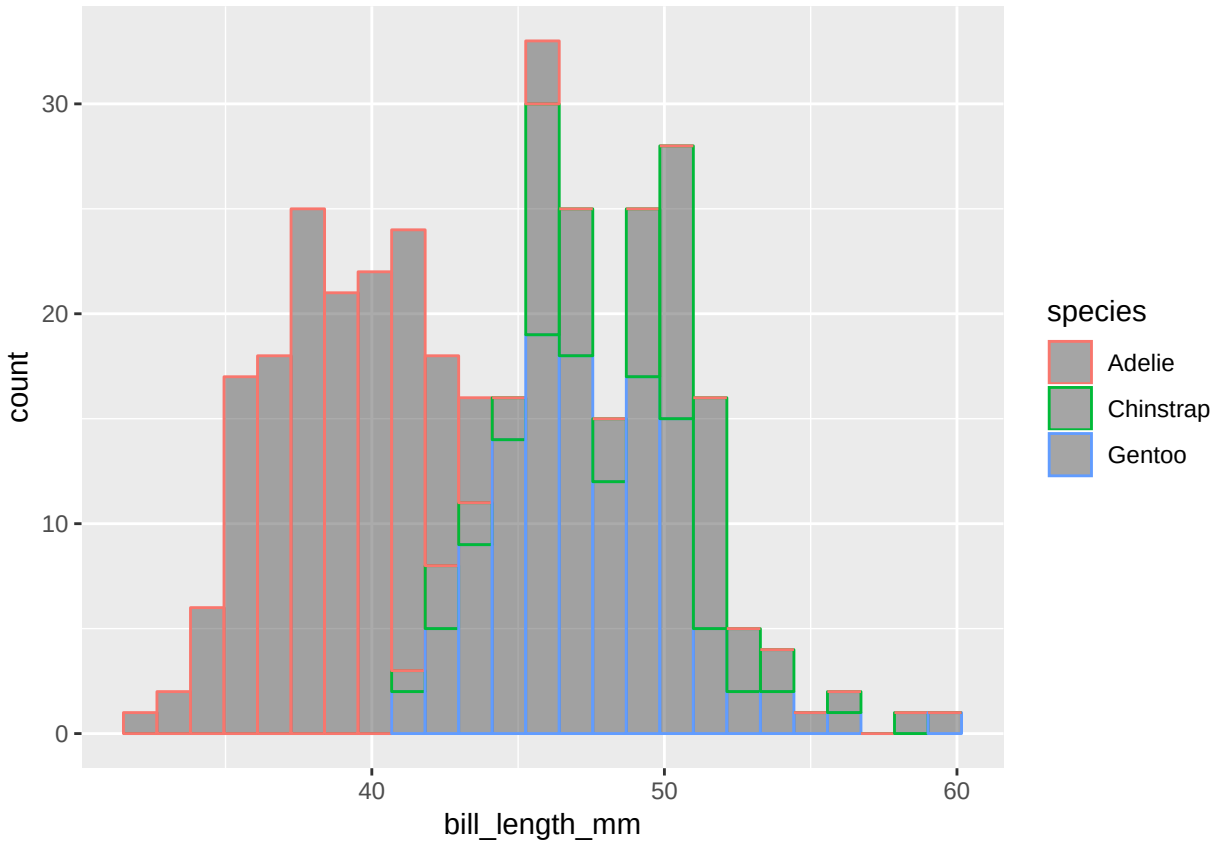
```
gf_histogram( ~ bill_length_mm, data = pingviner)
```



- How to make a histogram for some variable `x`:
  - Divide the interval from the minimum value of `x` to the maximum value of `x` in an appropriate number of equal sized sub-intervals.
  - Draw a box over each sub-interval with the height being proportional to the number of observations in the sub-interval.

- Not great for comparing groups:

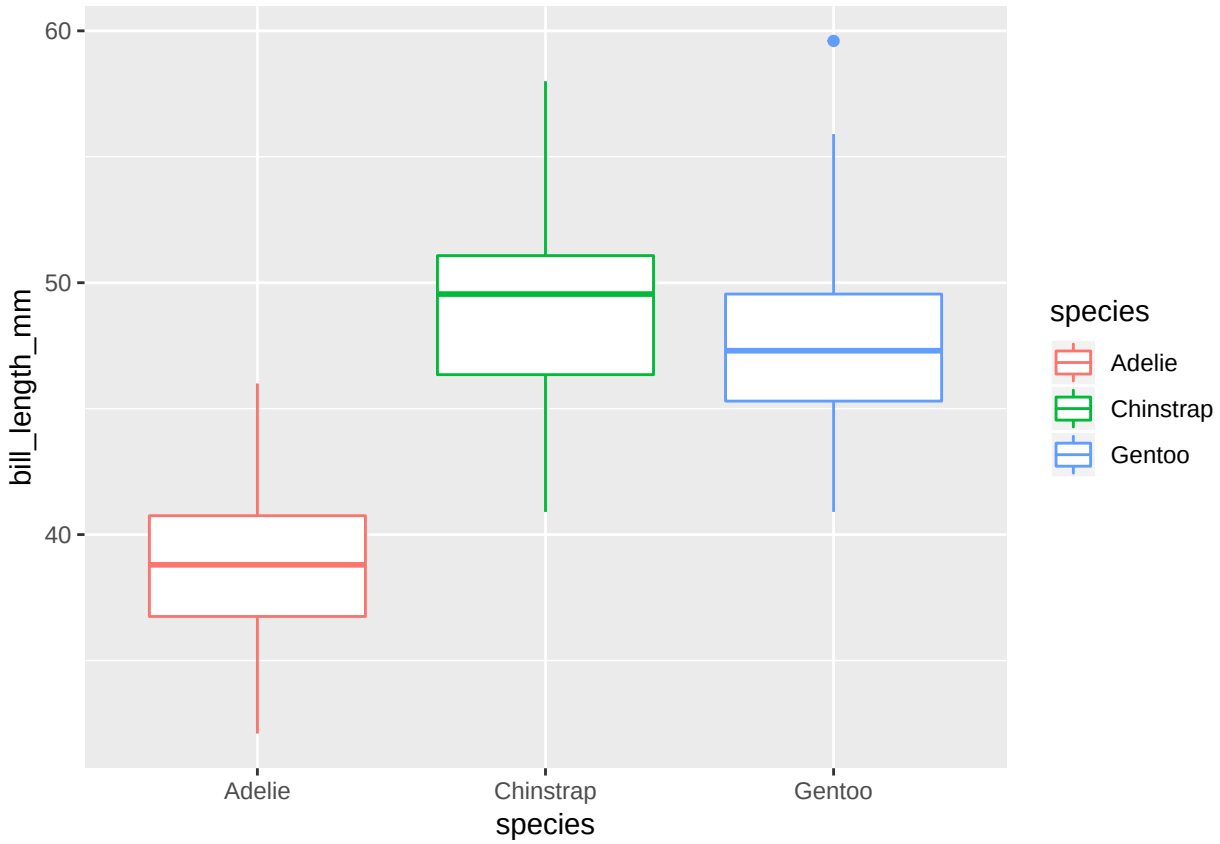
```
gf_histogram( ~ bill_length_mm, color = ~species, data = pingviner)
```



3.3 Boxplot

Boxplot can be good for comparing groups (notice we put the values on the y-axis here as it is more conventional for boxplots):

```
gf_boxplot(bill_length_mm ~ species, color = ~ species, data = pingviner)
```



To understand the details of the boxplot we need to introduce **percentiles**/quantiles and in particular quartiles which can be seen here:

```
Q <- quantile(bill_length_mm ~ species, data = pingviner, na.rm = TRUE)
Q
```

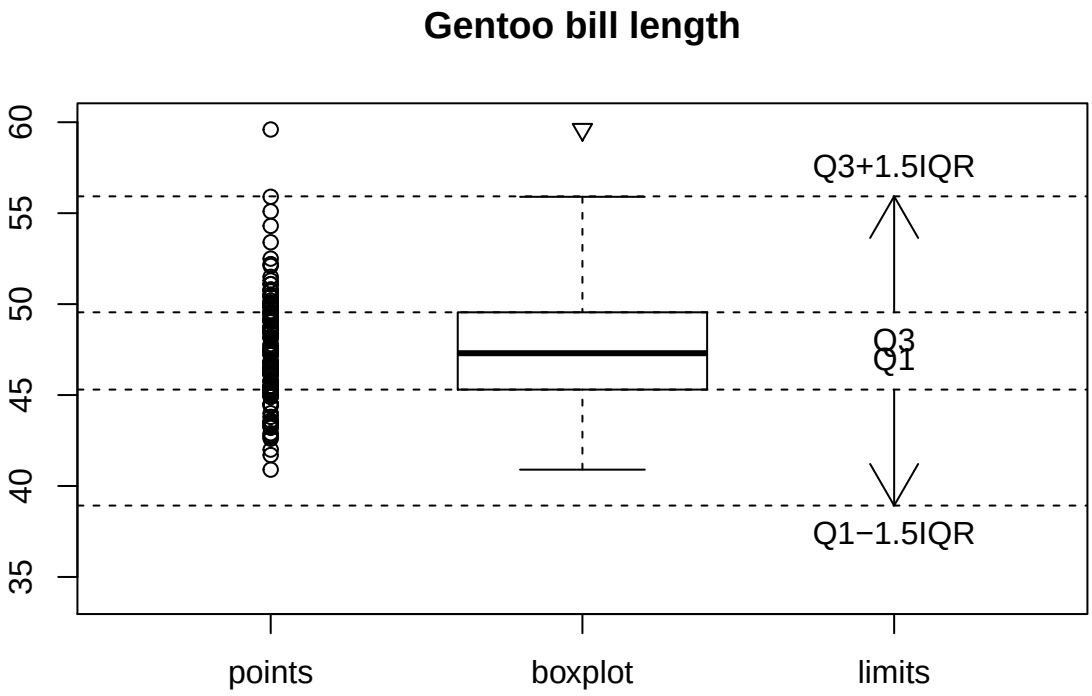
```
##     species 0% 25% 50% 75% 100%
## 1    Adelie 32  37  39  41   46
## 2 Chinstrap 41  46  50  51   58
## 3    Gentoo 41  45  47  50   60
```

- 50-percentile is the **median** and it is a measure of the center of data.
- 0-percentile is the **minimum** value.
- 25-percentile is called the **lower quartile** (Q1). Median of lower 50% of data.
- 75-percentile is called the **upper quartile** (Q3). Median of upper 50% of data.
- 100-percentil is the **maximum** value.
- **Interquartile Range (IQR)**: a measure of variability given by the difference of the upper and lower quartiles:
- Details about how to find/calculate percentiles are postponed to later.

3.3.1 How to draw a box plot

- Box:
  - Calculate the median, lower and upper quartiles.
  - Plot a line by the median and draw a box between the upper and lower quartiles.
- Whiskers:
  - Calculate interquartile range and call it IQR.
  - Calculate the following values:
    - L = lower quartile - 1.5*IQR
    - U = upper quartile + 1.5*IQR
  - Draw a line from lower quartile to the smallest measurement, which is larger than L.
  - Similarly, draw a line from upper quartile to the largest measurement which is smaller than U.
- Outliers: Measurements smaller than L or larger than U are drawn as circles.

Note: Whiskers are minimum and maximum of the observations that are not deemed to be outliers.



4 Data wrangling

4.1 Selecting columns/variables

- To subset columns of data use `select()` (automatically loaded from `dplyr` package by `mosaic`):

```
# Assigning the data subset to an object
bill_data <- select(pingviner, bill_length_mm, bill_depth_mm,
                    species, sex, island, year)

# This particular subset can be written shorter by (overwriting the object):
bill_data <- select(pingviner, -flipper_length_mm, -body_mass_g)
```

- Special role of first argument gives rise to this "pipe" (%>%) syntax:

```
bill_data <- pingviner %>% select(-flipper_length_mm, -body_mass_g)
```

- We read this as: first take the dataset **pingviner and then** select all columns except flipper_length_mm and body_mass_g.
- The resulting dataset doesn't have flipper length and body mass:

```
names(bill_data)
```

```
## [1] "species"        "island"         "bill_length_mm" "bill_depth_mm"
## [5] "sex"            "year"
```

4.2 Filtering rows/cases

- We use filter() to subset rows/cases. E.g. all penguins from Biscoe islands:

```
pingviner %>% filter(island == "Biscoe")
```

```
## # A tibble: 168 x 8
##    species island bill_length_mm bill_depth_mm flipper~ body_~ sex    year
##    <fctr>  <fctr>          <dbl>         <dbl>    <int>  <int> <fct> <int>
##  1 Adelie  Biscoe          37.8           18.3      174   3400 fema~  2007
##  2 Adelie  Biscoe          37.7           18.7      180   3600 male   2007
##  3 Adelie  Biscoe          35.9           19.2      189   3800 fema~  2007
##  4 Adelie  Biscoe          38.2           18.1      185   3950 male   2007
##  5 Adelie  Biscoe          38.8           17.2      180   3800 male   2007
##  6 Adelie  Biscoe          35.3           18.9      187   3800 fema~  2007
##  7 Adelie  Biscoe          40.6           18.6      183   3550 male   2007
##  8 Adelie  Biscoe          40.5           17.9      187   3200 fema~  2007
##  9 Adelie  Biscoe          37.9           18.6      172   3150 fema~  2007
## 10 Adelie  Biscoe          40.5           18.9      180   3950 male   2007
## # ... with 158 more rows
```

- All male Gentoo penguins with over 220 mm flippers:

```
pingviner %>% filter(sex == "male") %>%
  filter(species == "Gentoo") %>%
  filter(flipper_length_mm>220)
```

```
## # A tibble: 34 x 8
##    species island bill_length_mm bill_depth_mm flipper~ body_~ sex    year
##    <fctr>  <fctr>          <dbl>         <dbl>    <int>  <int> <fct> <int>
##  1 Gentoo  Biscoe          50.0           16.3      230   5700 male   2007
##  2 Gentoo  Biscoe          49.2           15.2      221   6300 male   2007
##  3 Gentoo  Biscoe          48.7           15.1      222   5350 male   2007
##  4 Gentoo  Biscoe          47.3           15.3      222   5250 male   2007
```

```
##  5 Gentoo  Biscoe           59.6          17.0      230   6050 male   2007
##  6 Gentoo  Biscoe           49.6          16.0      225   5700 male   2008
##  7 Gentoo  Biscoe           50.5          15.9      222   5550 male   2008
##  8 Gentoo  Biscoe           50.5          15.9      225   5400 male   2008
##  9 Gentoo  Biscoe           50.1          15.0      225   5000 male   2008
## 10 Gentoo  Biscoe           50.4          15.3      224   5550 male   2008
## # ... with 24 more rows
```

- This could also have been done with a single `filter()` command (output not shown):

```
pingviner %>% filter(sex == "male" & species == "Gentoo" & flipper_length_mm>220)
```

- All penguins of species Gentoo or Adelie:

```
pingviner %>% filter(species == "Gentoo" | species == "Adelie")
```

```
## # A tibble: 276 x 8
##    species island    bill_length_mm bill_depth_mm flipp~ body~ sex    year
##    <fctr>  <fctr>             <dbl>         <dbl>  <int> <int> <fct> <int>
##  1 Adelie  Torgersen           39.1          18.7    181  3750 male   2007
##  2 Adelie  Torgersen           39.5          17.4    186  3800 fema~  2007
##  3 Adelie  Torgersen           40.3          18.0    195  3250 fema~  2007
##  4 Adelie  Torgersen           NA            NA       NA    NA <NA>   2007
##  5 Adelie  Torgersen           36.7          19.3    193  3450 fema~  2007
##  6 Adelie  Torgersen           39.3          20.6    190  3650 male   2007
##  7 Adelie  Torgersen           38.9          17.8    181  3625 fema~  2007
##  8 Adelie  Torgersen           39.2          19.6    195  4675 male   2007
##  9 Adelie  Torgersen           34.1          18.1    193  3475 <NA>   2007
## 10 Adelie  Torgersen           42.0          20.2    190  4250 <NA>   2007
## # ... with 266 more rows
```

- This would be the same as penguins which are not Chinstrap (output not shown):

```
pingviner %>% filter(species != "Chinstrap")
```

4.3 Arranging rows/cases

- We use `arrange()` to arrange the order of the rows/cases:

```
pingviner %>% filter(sex == "female") %>% arrange(body_mass_g)
```

```
## # A tibble: 165 x 8
##    species   island    bill_length_mm bill_de~ flipper~ body_~ sex    year
##    <fctr>    <fctr>             <dbl>    <dbl>    <int>  <int> <fct> <int>
##  1 Chinstrap Dream               46.9     16.6      192   2700 fema~  2008
##  2 Adelie    Biscoe              36.5     16.6      181   2850 fema~  2008
##  3 Adelie    Biscoe              36.4     17.1      184   2850 fema~  2008
```

```
##  4 Adelie    Biscoe              34.5     18.1      187   2900 fema~  2008
##  5 Adelie    Dream               33.1     16.1      178   2900 fema~  2008
##  6 Adelie    Torgersen           38.6     17.0      188   2900 fema~  2009
##  7 Chinstrap Dream               43.2     16.6      187   2900 fema~  2007
##  8 Adelie    Biscoe              37.9     18.6      193   2925 fema~  2009
##  9 Adelie    Dream               37.0     16.9      185   3000 fema~  2007
## 10 Adelie    Dream               37.3     16.8      192   3000 fema~  2009
## # ... with 155 more rows
```

- Use `arrange(desc())` for descending values:

```
pingviner %>% filter(sex == "female") %>% arrange(desc(body_mass_g))
```

```
## # A tibble: 165 x 8
##    species island bill_length_mm bill_depth_mm flipper~ body_~ sex    year
##    <fctr>  <fctr>          <dbl>         <dbl>    <int>  <int> <fct> <int>
##  1 Gentoo  Biscoe           46.5          14.8      217   5200 fema~  2008
##  2 Gentoo  Biscoe           45.2          14.8      212   5200 fema~  2009
##  3 Gentoo  Biscoe           49.1          14.8      220   5150 fema~  2008
##  4 Gentoo  Biscoe           44.9          13.3      213   5100 fema~  2008
##  5 Gentoo  Biscoe           45.1          14.5      207   5050 fema~  2007
##  6 Gentoo  Biscoe           45.1          14.5      215   5000 fema~  2007
##  7 Gentoo  Biscoe           42.9          13.1      215   5000 fema~  2007
##  8 Gentoo  Biscoe           50.5          15.2      216   5000 fema~  2009
##  9 Gentoo  Biscoe           47.2          15.5      215   4975 fema~  2009
## 10 Gentoo  Biscoe           42.6          13.7      213   4950 fema~  2008
## # ... with 155 more rows
```