Manipulate and tidy data on R

Explore your data

1. Retrieve the data you read (three datasets corresponding to the sheets of phenotype.xlsx)

```
# library(readxl)
# data_phenotype <- read_xlsx("exercises/data/phenotype.xlsx", na = "NOT MEASURED")
```

2. Load `dplyr` and explore the three datasets using `glimpse`

Now some insights, this is data I have to manipulate during my PhD. I modified it, and shortened it in order to make things easier. But as it is, it is something looking almost like real life situations.

For this exercise, we will work on the sheet "plot" of the data.

It contains values of biomass (grain (=seed) and shoot parts) of 2 different species (barley, pea). The first column `code_crop` giving

1. crop_type (here just sole crops)
2. a letter for the species (B for barley, P for pea), a code for the cultivar (B1 for cultivar 1 for barley), the N treatment (N0 for unfertilized crop, N1 for fertilized crop) and a P treatment (here always P0, i.e unfertilzed crops).

4.1 Create a dataset (name it `data_barley`) containing only the observations of the barley, using the function `filter` of `dplyr`, select all columns except `x` and `y` and arrange them by `code_crop` and `block`.

4.2 Create a dataset (name it `data_pea`) containing only the observations of the pea, using the function `filter` of `dplyr`, select all columns except `x` and `y` and arrange them by `code_crop` and `block`.

These operations have to be done in two commands (one for barley, one for pea, without intermediate datasets).

Try to write it using the pipe and do it **without** to see the difference. Remember ctrl + shift + M to print it.

Hint : To `filter` rows containing observations of the barley or of the pea, you may need to use the `grepl` function, applied on the `cultivar` column. You may also use the `stringr::str_detect` function (tidyverse alternative to `grepl`).

5. The shoot parts of a plant contain the grain. We may need to compute the biomass of all shoot parts, *except* the seeds. Create a new column using `mutate` (name it biomass_no_seed) that is the difference between the biomass_shoot and the biomass_seed.

Hint: To avoid trouble while typing column name, you can use the Rstudio keyboard shortcut ctrl + space, it should suggest you the name of the columns of the dataset (**if you use the pipe**).

6. Suppose now we want to know how many treatments (i.e code_crop) are available for each cultivar. Do it using the `distinct` function.

What are the different treatments available for the barley and the pea ?

7. Suppose we want to see if the block number has an influence on the yield of the pea (biomass_seed). Print `data_pea`. What is the type of the block column ? How could you change it using `mutate` and `as.factor` ?

Carry out the following step to test it (using the %>%)

1. Take `data_pea`
2. Rename "biomass_seed.2002-08-08" to Yield (function `rename` of dplyr)
3. Change the type of the block column using the function `mutate`
4. Run an ANOVA (think on what you should put in the formula)
5. Print a summary

Conclude

In practice, we should verify the ANOVA assumptions and study the residuals, but it is not what matters here :-) .

8. Compute the mean and the standard deviation *by cultivar* of the yield for the barley and summarise it in a table

9. Use `across` function to transform biomass' values in kg/ha instead of t/ha for the barley's values.

Try to do it using `where(is.numeric)` *and* `starts_with("biomass")` as `.cols` argument (see `?across`).

10. Use `left_join` function between `data_barley` and `data_itk` to get information about the density_level and the cultivar of the barleys cultivated. How many different barley cultivars are present in the data base ? **Hint** Use `distinct` function after (= %>%) the `left_join` to get the information asked.

Do the same for the pea.

Tidy your data

1. Load `tidyr`

2. Use the function `bind_rows` to bind `data_barley` and `data_pea` (simply `bind_rows(data_pea, data_barley)`. Put the result in a dataset called `data_species`

3. Use `pivot_longer` to gather all columns starting with "biomass" into a new column called "trait" (like a functional trait). Print the new dataset

4. Use `separate` (see `?separate`) to turn the column `trait` into two new columns : The first called "biomass_type", the second called "date".

Hint: You need to specify `sep = "\\."` (it is a regular expression)

5. Use `pivot_wider` to spread your type of biomass into new columns (do not spread the `date` column !)

6. **If you have the time**:

Print your new dataset, what is the type of `date` ? Is it convenient ?

Run the following command: `data_biomass %>% mutate(date = lubridate::ymd(date))` (replace data_biomass by the name of your dataset).

What does it change ?

In practice, it is more convenient to work with date objects than with characters.