ADS2 Practical 7: Getting and cleaning Data

Dmytro Shytikov and Xushen Xiong

2023-11-02

In this practical session, you will work on two real-world datasets that are modified to suit our practical purpose. You could work through this guide alone or in groups. Facilitators are here to help. The time it takes to complete this practical can vary between individuals – this is OK. Do not worry if you do not finish within the session.

Learning Objectives

- Getting data in R
- Use different data types and data structures in R
- Explain advantages of tidy datasets
- Clean a real-world dataset according to tidy principles

Help tips:

Any functions that you want to know more/better, use `? Function_name`. There are many useful libraries in R that makes data cleaning much more efficient and easier.

Practice 1. West Nile Virus (WNV) Mosquito Test Background

A short summary of the data:

This data shows locations and test results for pools of mosquitoes through the Chicago Department of Public Health Environmental Health program. The Chicago Department of Public Health maintains an environmental surveillance program for West Nile Virus (WNV). This program includes the collection of mosquitoes from traps located throughout the city; the identification and sorting of mosquitoes collected from these traps; and the testing of specific species of mosquitoes for WNV.

Source: data.cityofchicago.org

This dataset is trimmed and modified from the original dataset for the practice purpose.

- Getting the data

Input the `WNV_mosquito_test_results.csv` using `read.csv ()`.
Remember to set the working directory `setwd()` and specify the path to the file.

- Explore your data. `head(), tail(), nrow() , attributes(), table()`
- Data types and the structure?

1. What is the data structure? `str()`
2. What data types do we have? `class()`

- Is this a tidy dataset?

3. Screen and diagnosis.

- Is this a long or wide form?
- Are the variable names informative and precise? (**Hint**: The name of the first variable `SEASON.YEAR` is not accurate. Please change it to `YEAR`.)
- Are there missing values? `anyNA()`
- Are there duplicated values?
- Are there any strange patterns? You will need to make different diagnostic plots to spot any strange patterns. Try to be creative!

4. Treat and document.

- Do we need to convert the data structure from long to wide and vice versa?
- Are the variable names informative and precise? (**Hint**: The name of the first variable `SEASON.YEAR` is not accurate. Please change it to `YEAR`.)
- Shall we remove missing values? Some useful functions are `complete.cases()`, `drop_na()`.
- Shall we remove duplicated values?
- Shall we remove any additional data?

Special about this data: What is the datatype of the variable `TEST.DATE`? There is specific datatype to manipulate date/time, please try `as.POSIXct()` to convert them, be careful of the `format` and `tz` arguments. (**Try**: the time zone of Chicago is `America/Chicago`. Check the change of the class afterwards. Assign the first date/time to `dat1`, check the attributes of `POSIXct datatype`. Then alter the time zone attribute to `America/Los_Angeles`, and see what happens to `dat1`.)

The `LOCATION` variable consists of two elements `LATITUDE` and `LONGITUDE`, try command `gsub()` to remove `and` first and then use command `separate` to generate two new variables based on `LOCATION` (**Tip**: Try settings in arguments `remove` and `convert` to see different outputs).

Remember to document all the changes that you applied to the original data. Also state the reasoning behind them.

- What is your data telling you?

5. Summarize your data:

- Is there any relationship between the number of mosquitoes caught and the year?
- Is there any relationship between the location, longitude, or latitude, and the number of mosquitoes caught?
- Some types of traps may catch a particular types of mosquitoes more often. Can you see it from the data?
- Formulate any other hypothesis.

Try an appropriate graph type to fit your data best. Use either in-built plotting functions (`base` package) or those from `ggplot2` package.

Practice 2. Tests for antibodies to trachoma PGP3 antigen

A short summary of the data:

This set includes data used in a latent class model to compare testing platforms for detection of antibodies against the *Chlamydia trachomatis* antigen *Pgp3*.

Source: data.cdc.gov

The dataset is trimmed from original dataset.

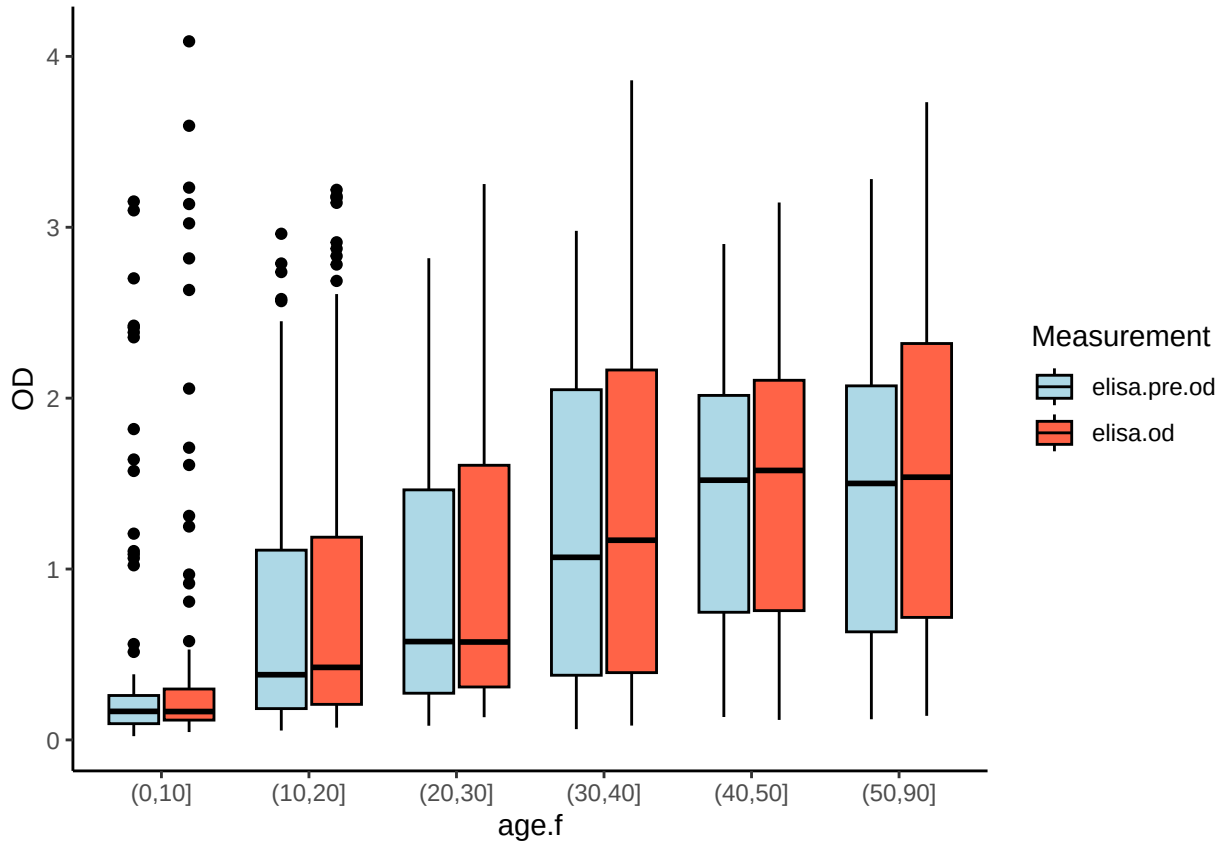
- Getting the data

Input the `Tests_PGP3.txt` (**Hint**: Try different input functions or parameter settings. Do the input data formats look the same?)

- Explore, diagnose and clean the data to the standard of the tidy data.
  - Is this a long or wide form?
  - Do we need to convert it?
- Is this a tidy dataset?
- Screen and diagnosis.
  - Is this a long or wide form? Explore the functions `gather()`, `spread()`
  - Do we need to convert it? Try to stick to the "tidy data" principle.
  - Are the variable names informative and precise? (**Hint**: The variable `sex` is not readable, convert to more readable format. Men are coded as `1` and women are coded as `2`).
  - Are there missing values? Pay attention to the types of missing values.
  - Are there duplicated values? Try `which( duplicated() )` . `which()` returns the index. Try to set the parameter `fromLast` as `TRUE` in the function `duplicated()`
  - Are there any strange patterns?
- Treat and document.
  - Explore the functions `gather(), spread()` to reshape the data. Explain why you decided to do that way.
  - Are the variable names informative and precise?
  - Shall we remove missing values? Some useful functions are `complete.cases(), drop_na()`.
  - Shall we remove duplicated values?
  - Shall we remove any additional data?
- Let's examine the relationship between `ELISA.od` and `age`.

If you haven't done yet, please reshape the data to combine `elisa.od` and `elisa.pre.od` values since they are both ELISA measurements, but at different time points. Try to use command `gather()` in `tidyr` package to reshape the data frame so that the two measurements are combined into one variable `ELISA.od` (`key="time.point"`, try the argument `"factor_key"`).

Use `ggplot2` to plot a boxplot to view the relationships between `ELISA.od` and `age.f`, use `color` argument to group `time.point`. You should see a plot like this.



Originally, created by Chaochen Wang in 2019.

Last update by D Shytikov in 2023.