Session 1: Introduction to data analysis in R - a practical example

Christoph Schürz *christoph.schuerz@boku.ac.at*

The goal of the first session is that you get familiar with simple data analysis tasks in *R*. We learn basic tasks such as loading data, working with tables and visualizing analysis results, using Riesel climatic data.

Contents

Required packages

```
library(tidyverse)
library(pasta)
library(here)
```

Data acquisition

Loading climatic data directly from a webpage

I added this section just for the sake of completeness. It is a great example for all the challanges "real" data pose when working with them. Below I provide some ideas about working with "real and messy" data and how we can do better. I think this is important for working with data and it might spare you some hassle in the future. It is not essential to be able to follow every step of this section. You can download the resulting cleaned up dataset from the github repository. We will use that dataset to learn how to load data from your hard drive in the following section.

The Riesel climatic data is available from the ARS webpage. In general, there is not much of a difference between a path to a file on your comuter and a URL. Therefore, we can directly load the data from the internet.

```
# The URL where the weather files on the ARS webpage are stored
ars_url <- "https://www.ars.usda.gov/ARSUserFiles/30980000/riesel/climate"
# A verctor of the names of the weather files
weather_files <- 2010:2012%&%"hrly.txt"

# Loop over all weather file names, load the files from the web page and
# merge them to a tibble (data.frame)
ars_clim <- map_dfr(weather_files, ~read_table(file = url(ars_url%//%.x),
                                               guess_max =  10000,
                                               skip = 3, col_names = F))
```

The header of the tables provided online are (no offense) a bad example of storing data. Therefore I skipped the header line `skip = 3` when loading the data. In order to give the variables good names (one of the hardest tasks in programming) to work with in the following I created a name vector with "good" variable names. We will discuss the naming of variables and I will share some of my ideas to this topic at this point.

```
# The variable names we will assign to the variables in the weather data set.
tbl_header <- c("year",         # yyyy
                "day",          # jdn
                "hour",         # (h)hmm
                "t_air_ave",    # degC
                "t_air_max",    # degC
                "t_air_min",    # degC
                "rh_max",       # %
                "rh_min",       # %
                "p_vap_ave",    # kPa
                "sr_tot",       # kJ m^-2
                "wnd_v_ave",    # m s^-1
                "wnd_v_max",    # m s^-1
                "wnd_dir_ave", # deg
                "pr_tot",       # mm
                "t_sol_ave",    # degC
                "t_sol_max",    # degC
                "t_sol_min"     # degC
                )

names(ars_clim) <- tbl_header

ars_clim
```

```
## # A tibble: 29,679 x 17
##     year   day  hour t_air_ave t_air_max t_air_min rh_max rh_min p_vap_ave
##    <dbl> <dbl> <dbl>     <dbl>     <dbl>     <dbl>  <dbl>  <dbl>     <dbl>
##  1  2010     1   100      2.08       2.75      1.62   81.0   78.5     0.570
##  2  2010     1   200      1.28       1.68      0.88   83.8   80.8     0.55
##  3  2010     1   300      0.67       1.01      0.34   85.4   83.6     0.54
##  4  2010     1   400      0.08       0.48     -0.25   86.9   85.2     0.53
##  5  2010     1   500    -0.580     -0.18     -0.99   88.8   86.6     0.51
##  6  2010     1   600    -1.11      -0.78     -1.45   89.1   88.0     0.5
##  7  2010     1   700    -1.32      -1.18     -1.45   89.4   87.8     0.49
##  8  2010     1   800    -1.36      -1.24     -1.51   88.0   86.0     0.48
##  9  2010     1   900    -0.7        0.16     -1.31   86.2   80.5     0.49
## 10  2010     1  1000     1.03       1.9      -0.04   80.7   70.9     0.5
## # ... with 29,669 more rows, and 8 more variables: sr_tot <dbl>,
## #   wnd_v_ave <dbl>, wnd_v_max <dbl>, wnd_dir_ave <dbl>, pr_tot <dbl>,
## #   t_sol_ave <dbl>, t_sol_max <dbl>, t_sol_min <dbl>
```

Saving data

There are many ways and data formates to save your data. If I do not intend to use the data with any other software (e.g. Excel) or share the dataset with colleagues, my preferred way to store data is as ***.rds** files.

```
saveRDS(ars_clim, here("Session_1/ars_clim.rds"))
```