Data Import

David Gerard

2019-04-02

Learning Objectives

- Import data from CSV's,
- Working Directories
- Chapter 11 of RDS
- Data Import Cheat Sheet.
- Readr Overview.

Working Directories

- The working directory is where R will look for and save things by default.

- When you specify to save a figure, save a file, or load some data, it will be with respect to the working directory.

- You can see where the current working directory is by `getwd()`, or by looking at the top of the console in RStudio.

- You can change the working directory by Session > Set Working Directory > Choose Directory. Or by CONTROL + SHIFT + H. Or you can use the `setwd()` command.

- A shortcut is to set the working directory to your source file location with Session > Set Working Directory > To Source File Location.

- When you read and write files/figures, you can then specify the path from the position of the working directory.

- Suppose we want to save the following figure:

```
suppressPackageStartupMessages(library(tidyverse))
data("mpg")
pl <- ggplot(mpg, aes(x = hwy, y = cty)) +
  geom_point()
```

- To save `pl` in the current folder, we would use:

```
ggsave(filename = "./my_saved_plot.pdf", plot = pl)
```

- The "`.`" means "the current folder".

- To save `pl` in the folder one level up we would use:

```
ggsave(filename = "../my_saved_plot.pdf", plot = pl)
```

- The "`..`" means "go one level up".

- If we are in the analysis folder, and we want to save `pl` in the output folder, we would use:

```r
ggsave(filename = "../output/my_saved_plot.pdf", plot = pl)
```

- If we have a subfolder called “fig” within out current folder. We could save `pl` in “fig” with

```r
ggsave(filename = "./fig/my_saved_plot.pdf", plot = pl)
```

- **NEVER USE ABSOLUTE PATHS**. For example, you should never start the path from “C” if you use Windows. This makes your code non-transferable to other users.

readr

- A lot of datasets come in comma-separated or tab-separated formats. For example, These are the first few rows of hate_crimes2.csv (available at https://dcgerard.github.io/stat_412_612/data.html):

```
state,median_house_inc,share_unemp_seas,share_pop_metro,share_pop_hs,share_non_citizen,share_white_
Alabama,42278,0.06,0.64,0.821,0.02,0.12,0.472,0.35,0.63,0.125838926,1.806410489
Alaska,67629,0.064,0.63,0.914,0.04,0.06,0.422,0.42,0.53,0.143740118,1.656700109
Arizona,49254,0.063,0.9,0.842,0.1,0.09,0.455,0.49,0.5,0.225319954,3.413927994
Arkansas,44922,0.052,0.69,0.824,0.04,0.12,0.458,0.26,0.6,0.069060773,0.869208872
California,60487,0.059,0.97,0.806,0.13,0.09,0.471,0.61,0.33,0.255805361,2.397985899
Colorado,60940,0.04,0.8,0.893,0.06,0.07,0.457,0.31,0.44,0.390523301,2.804688765
Connecticut,70161,0.052,0.94,0.886,0.06,0.06,0.486,0.3,0.41,0.335392269,3.772701469
Delaware,57522,0.049,0.9,0.874,0.05,0.08,0.44,0.37,0.42,0.322754169,1.469979563
District of Columbia,68277,0.067,1,0.871,0.11,0.04,0.532,0.63,0.04,1.52230172,10.95347971
```

- In the file, each column is separated by a comma. Each row is separated by a new line.
- We will use the readr package to load these datasets into R.
- The readr package is a part of the tidyverse, and so it is automatically loaded when you load the tidyverse.
- To read a CSV (comma-separated values) file into R, use the `read_csv()` function from the readr package.

```r
library(tidyverse)
hate_crimes <- read_csv(file = "../../data/hate_crimes1.csv")
```

```
## Parsed with column specification:
## cols(
##   `state median_house_inc     share_unemp_seas     share_pop_metro share_pop_hs     share_non_citiz
## )
```

- If the CSV is online and you know the URL, you can use that URL for the `file` argument.

```r
library(tidyverse)
hate_crimes <- read_csv(file = "https://dcgerard.github.io/stat_412_612/data/hate_crimes2.csv")
```

```
## Parsed with column specification:
## cols(
##   state = col_character(),
##   median_house_inc = col_double(),
##   share_unemp_seas = col_double(),
##   share_pop_metro = col_double(),
##   share_pop_hs = col_double(),
##   share_non_citizen = col_double(),
##   share_white_poverty = col_double(),
##   gini_index = col_double(),
##   share_non_white = col_double(),
##   share_vote_trump = col_double(),
##   hate_crimes_per_100k_splc = col_double(),
##   avg_hatecrimes_per_100k_fbi = col_double()
## )
```

- Use `read_tsv()` if columns are separated by tabs. (if you use `read_lines()` on a tsv, the tabs will show up as "\t").
- Use `read_csv2()` if columns are separated by semicolons.
- Other file formats are listed in RDS.
- You want to import data directly from Excel? Don't.
    - First export the Excel spreadsheet as a CSV. Then read the CSV file into R.
- You are using colors to represent meaningful information in Excel? Don't.
    - Edit the data so that the information is encoded by a new variable.
- If you don't know the format ahead of time, use `read_lines()` to print the first few lines.

```
read_lines(file = "../../data/hate_crimes2.csv", n_max = 10)
```

```
##  [1] "state,median_house_inc,share_unemp_seas,share_pop_metro,share_pop_hs,share_non_citizen,sha
##  [2] "Alabama,42278,0.06,0.64,0.821,0.02,0.12,0.472,0.35,0.63,0.125838926,1.806410489"
##  [3] "Alaska,67629,0.064,0.63,0.914,0.04,0.06,0.422,0.42,0.53,0.143740118,1.656700109"
##  [4] "Arizona,49254,0.063,0.9,0.842,0.1,0.09,0.455,0.49,0.5,0.225319954,3.413927994"
##  [5] "Arkansas,44922,0.052,0.69,0.824,0.04,0.12,0.458,0.26,0.6,0.069060773,0.869208872"
##  [6] "California,60487,0.059,0.97,0.806,0.13,0.09,0.471,0.61,0.33,0.255805361,2.397985899"
##  [7] "Colorado,60940,0.04,0.8,0.893,0.06,0.07,0.457,0.31,0.44,0.390523301,2.804688765"
##  [8] "Connecticut,70161,0.052,0.94,0.886,0.06,0.06,0.486,0.3,0.41,0.335392269,3.772701469"
##  [9] "Delaware,57522,0.049,0.9,0.874,0.05,0.08,0.44,0.37,0.42,0.322754169,1.469979563"
## [10] "District of Columbia,68277,0.067,1,0.871,0.11,0.04,0.532,0.63,0.04,1.52230172,10.95347971'
```

Special Considerations

- **Always check your data immediately after importing it**.
    - Check that the types are correct for each of the variables.
    - Check that the missing data were coded correctly.
    - Later on, when you notice something weird, consider that this might have resulted because of a problem during data import.

```r
hate_crimes %>%
  summarize_all(class)
```

```
## # A tibble: 1 x 12
##   state median_house_inc share_unemp_seas share_pop_metro share_pop_hs
##   <chr> <chr>            <chr>            <chr>           <chr>
## 1 char~ numeric          numeric          numeric         numeric
## # ... with 7 more variables: share_non_citizen <chr>,
## #   share_white_poverty <chr>, gini_index <chr>, share_non_white <chr>,
## #   share_vote_trump <chr>, hate_crimes_per_100k_splc <chr>,
## #   avg_hatecrimes_per_100k_fbi <chr>
```

```r
hate_crimes %>%
  summarize_all(funs(sum(is.na(.))))
```

```
## Warning: funs() is soft deprecated as of dplyr 0.8.0
## please use list() instead
##
## # Before:
## funs(name = f(.)
##
## # After:
## list(name = ~f(.))
## This warning is displayed once per session.
```

```
## # A tibble: 1 x 12
##   state median_house_inc share_unemp_seas share_pop_metro share_pop_hs
##   <int>            <int>            <int>           <int>        <int>
## 1     0                0                0               0            0
## # ... with 7 more variables: share_non_citizen <int>,
## #   share_white_poverty <int>, gini_index <int>, share_non_white <int>,
## #   share_vote_trump <int>, hate_crimes_per_100k_splc <int>,
## #   avg_hatecrimes_per_100k_fbi <int>
```

```r
head(hate_crimes)
```

```
## # A tibble: 6 x 12
##   state median_house_inc share_unemp_seas share_pop_metro share_pop_hs
##   <chr>            <dbl>            <dbl>           <dbl>        <dbl>
## 1 Alab~            42278            0.06             0.64        0.821
## 2 Alas~            67629            0.064            0.63        0.914
## 3 Ariz~            49254            0.063            0.9         0.842
## 4 Arka~            44922            0.052            0.69        0.824
## 5 Cali~            60487            0.059            0.97        0.806
## 6 Colo~            60940            0.04             0.8         0.893
## # ... with 7 more variables: share_non_citizen <dbl>,
## #   share_white_poverty <dbl>, gini_index <dbl>, share_non_white <dbl>,
## #   share_vote_trump <dbl>, hate_crimes_per_100k_splc <dbl>,
## #   avg_hatecrimes_per_100k_fbi <dbl>
```

- Sometimes the files code missing data other than `NA`. For example, it's common to use periods `.`, or in some genomic settings they use `-9` as missing.

- R won't know how to handle this without you telling it, so you'll have to know what the missing data encoding is and specify it with the `na` argument in `read_csv()`.
- readr will try to guess the type for each column (double, integer, character, logic, etc). Sometimes it guesses wrong. If it seems to be guessing wrong, use the `col_types` to explicitly specify the column types.
- Sometimes there are comments at the start of a data file. You can skip the first few lines before starting to read data with the `skip` argument.
- If the comments begin with a special character, you can use the `comment` argument.
- **Exercise**: Successfully load all of the `hate_crimes` CSV files at https://dcgerard.github.io/stat_412_612/data.html.

Data Export

- You can write comma-separated and tab-separated files using `write_csv()`, `write_csv2()`, and `write_tsv()`.
- The defaults are usually fine.

Reading/Writing R Objects

- You can save and reload arbitrary R objects (data frames, matrices, lists, vectors) using `readRDS()` and `saveRDS()`.