Exercise 0: Introduction

Henrik Singmann

January 2019 (updated: 2019-05-19)

Freeman, Heathcote, Chalmers, and Hockley (2010) data

The data are lexical decision and word naming latencies for 300 words and 300 nonwords from 45 participants presented in Freeman et al. (2010). The 300 items in each `stimulus` condition were selected to form a balanced 2×2 design with factors neighborhood `density` (low versus high) and `frequency` (low versus high).

The `task` was a between subjects factor: 25 participants worked on the lexical decision task and 20 participants on the naming task. After excluding erroneous responses each participants responded to between 135 and 150 words and between 124 and 150 nonwords.

- Lexical decision task: Decide whether a string of letters presented on screen is a word (e.g., house) or a non-word (e.g., huese). Response times were recorded when participants pressed the corresponding response key (i.e., word or non-word).
- Naming task: Read the word presented in the screen. Response times were recorded when participants started saying the presented word.

Design

The data comes with package `afex`, so we can load it right away. But at first, we load the `tidyverse` package, because these are the functions we want to use throughout this exercise.

```
library("tidyverse")
data("fhch2010", package = "afex") # load
fhch <- fhch2010 %>%
  filter(correct) %>% # remove errors
  droplevels          # remove empty factor levels
str(fhch) # structure of the data
```

```
## 'data.frame':    12960 obs. of  10 variables:
##  $ id       : Factor w/ 45 levels "N1","N12","N13",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ task     : Factor w/ 2 levels "naming","lexdec": 1 1 1 1 1 1 1 1 1 1 ...
##  $ stimulus : Factor w/ 2 levels "word","nonword": 1 1 1 2 2 1 2 2 1 2 ...
##  $ density  : Factor w/ 2 levels "low","high": 2 1 1 2 1 2 1 1 1 1 ...
##  $ frequency: Factor w/ 2 levels "low","high": 1 2 2 2 2 2 1 2 1 2 ...
##  $ length   : Factor w/ 3 levels "4","5","6": 3 3 2 2 1 1 3 2 1 3 ...
##  $ item     : Factor w/ 600 levels "abide","acts",..: 363 121 202 525 580 135 42 368 227 141 ...
##  $ rt       : num  1.091 0.876 0.71 1.21 0.843 ...
##  $ log_rt   : num  0.0871 -0.1324 -0.3425 0.1906 -0.1708 ...
##  $ correct  : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
```

The columns in the data are:

- `id`: participant id, `factor`
- `task`: `factor` with two levels indicating which task was performed: `"naming"` or `"lexdec"`
- `stimulus`: `factor` indicating whether the shown stimulus was a `"word"` or `"nonword"`
- `density`: `factor` indicating the neighborhood density of presented items with two levels: `"low"` and `"high"`. Density is defined as the number of words that differ from a base word by one letter or phoneme.

- `frequency`: `factor` indicating the word frequency of presented items with two levels: `"low"` (i.e., words that occur less often in natural language) and `"high"` (i.e., words that occur more often in natural language).
- `length`: `factor` with 3 levels (4, 5, or 6) indicating the number of characters of presented stimuli.
- `item`: `factor` with 600 levels: 300 words and 300 nonwords
- `rt`: response time in seconds
- `log_rt`: natural logarithm of response time in seconds
- `correct`: boolean indicating whether or not the response in the lexical decision task was correct or incorrect (incorrect responses of the naming task are not part of the data).

Exercise 1: Calculating Simple Summary Measures

For this and the following exercises use the `fhch` `data.frame` (i.e., the data after removing errors).

Part A:

Use your knowledge of `dplyr` in combination with the pipe `%>%` and take the `mean` of the `rt` column, conditional on `task`. For which task are participants on average faster?

Hints:

- `group_by` can be used for conditioning on one or several variables. Separate more than one variable by comma.
- `summarise` can be used for aggregating multiple lines into one.
- The pipe `%>%` allows to concatenate calls from left to right (shortcut for the pipe is `ctrl/cmd` + `shift` + `m`).
- More information: https://github.com/rstudio/cheatsheets/raw/master/data-transformation.pdf

```
# start with:
#fhch %>% ...
```

Part B

`summarise` allows to use more than one aggregation function. Extend the previous code and also calculate the standard deviation, `sd()`, per task. Does the task that is faster also have the lower variability in RT (i.e., smaller sd)?

```
# fhch %>% ...
```

Part C

Means are quite sensitive to outliers. Therefore, please recalculate `mean` and `sd` per task, after removing some extreme outliers. Here, we define outliers as RTs below .25 seconds and above 2.5 seconds. Do we still find the same pattern?

Remember, the verb for selecting variables using `dplyr` is `filter`. You can concatenate various filters simply by comma, in the same call to `filter()`.

```
# fhch %>% ...
```

Exercise 2: Aggregating Data by ID and Plotting

The `fhch` data has multiple observations (i.e., trials) per participant and cell of the design. In a traditional analysis, for example, using ANOVA, one can only have one observation per participants and cell of the design. Therefore, a common task is to aggregate the data on the level of the participant and combinations of factors one is currently interested in.

Part A

Use the data from the `"lexdec"` task only. For this, take the `mean` of the `rt` column per participant and level of the `length` factor. Save this data in a new object `agg1`.

Note that to condition on more than one variable in `group_by()`, simply separate the variables by comma.

```
## write code here
```

Part B

Let us take a look at the individual-level data per length level that you just created. For this, use `ggplot` and plot the level of `length` on the x-axis and the mean RTs on the y-axis.

- Try both `geom_point` and `geom_jitter()`. Which looks better?
- Does this plot show any clear pattern?
- Can you think of a way to make this plot more informative?

```
## write code here
```

```
## write code here
```

Part C

Make a plot similar as above, but this time also condition on the `density` factor. That is, first aggregate the data again, this time for the combination of `id`, `length`, and `density`. Then plot the data as above, but also add an aesthetic for the `density` factor. Use `color` to distinguish the different levels of `density` in the plot. Can you see something in this plot? If not, have a look at `position_dodge` with `geom_point`.

```
## write code here
```

Ressources

- `RStudio` cheat sheets: https://www.rstudio.com/resources/cheatsheets/
  - `RStudio`: https://github.com/rstudio/cheatsheets/raw/master/rstudio-ide.pdf
  - `ggplot2`: https://github.com/rstudio/cheatsheets/raw/master/data-visualization-2.1.pdf
  - `dplyr` & `tidyr`: https://github.com/rstudio/cheatsheets/raw/master/data-transformation.pdf

References

- Freeman, E., Heathcote, A., Chalmers, K., & Hockley, W. (2010). Item effects in recognition memory for words. *Journal of Memory and Language*, 62(1), 1-18. https://doi.org/10.1016/j.jml.2009.09.004