Neuroinflammation effect on context discrimination memory: Data Import and Cleaning of raw training and testing behavioral data

G. Lewandowski

March 31, 2017

Contents

Document Introduction

In this document the raw behavioral training and testing data for experiments 14x02 and 14x04 are imported. The imported data will be converted (if needed) to a tidy data form. The data will be compiled such that there is one behavioral training dataset and one behavioral testing dataset.
Notably, for each dataset observation is synonymous with row and variable is synonymous with column. Most frequently, variable will be used instead of column.

Contextual Discrimination Conditioning overview

Contextual discrimination conditioning (CDC) memory was assessed by measuring freezing in two similar behavioral chambers. There was a mild footshock in context A but not in context A'.

For CDC training: on each training day the rats were place in each context for 3min 20s. For each min and for the final 20s interval the amount of freezing in seconds was recorded. The context order (AA' or A'A) was changed for each training day and was also recorded. Rats were were trained to criterion as defined by **>=25% freezing in A and <=10% freezing in A' for 2 consecutive days**.

For CDC testing: The testing data consist of one session in which the rats spent 3min 20s in context A' and then 5 min in context A. Again freezing (in seconds) was recorded for each minute interval.

Load R packages required for data import and cleaning

```r
suppressMessages({
    library(rmarkdown)  ## render markdown docs (output)
    library(knitr)  ## knit markdown docs (output)
    library(readxl) ## read MS Excel worksheets into data tables
    library(pander)  ## generate tables in output
    library(broom) ## 'tidy()' summary output into a data frame
    library(stringr)  ## use substr() to extract info
    library(tidyverse)  ## data wrangling: dplyr, tidyr
})
```

Data Import

The behavioral data for 14x02 is contained in two worksheets within a Microsoft Excel workbook. Likewise, the behavioral data for 14x04 is contained in two worksheets within a Microsoft Excel workbook. The 14x04 workbook also contains the context order for the training days (both 14x02 and 14x04). Finally, an additional workbook contains the treatment group assignments for the rats in both studies.
In this step the data is imported from the Excel workbook into data tables.

```r
## path to 14x02 Excel workbook with experiment data (base R)
path_14x02 <- file.path("neuroinflam_CDC_raw_data",
    "14x02_raw_ behav_data.xlsx")

## path to 14x04 Excel workbook with experiment data
path_14x04 <- file.path("neuroinflam_CDC_raw_data",
    "14x04_raw_ behav_data.xlsx")
```

```r
## path to the group assignments for 14x02 and 14x04
path_grps <- file.path("neuroinflam_CDC_raw_data",
    "group_assingments.xlsx")
```

The worksheets in the 14x02 Microsoft workbook are:

```r
## use the function excel_sheets() from the readxl package

## list the sheets in the 14x02 Excel workbook
excel_sheets(path_14x02)
```

```
## [1] "Freezing" "Test Day"
```

The worksheets in the 14x04 Microsoft workbook are:

```r
## list the sheets in the 14x04 Excel workbook
excel_sheets(path_14x04)
```

```
## [1] "Freezing"                    "Test Day"
## [3] "context_order"               "Context presentation order"
```

For both 14x02 and 14x04 the "Freezing" worksheets contain the behavioral training data. The "Test Day" worksheets contain the behavioral testing data. And the "context_order" worksheet of the 14x04 workbook contains the context order for each training day.

Import 14x02 behavioral data

```r
## use the read_excel() function from the readxl package

## import 14x02 training data
train_14x02 <- read_excel(path = path_14x02,
                              sheet = "Freezing",
                              na = "")

## import 14x02 testing data
test_14x02 <- read_excel(path = path_14x02,
                              sheet = "Test Day",
                              na = "")
```

Import 14x04 behavioral data

```r
## use the read_excel() function from the readxl package

## import 14x04 training data
train_14x04 <- read_excel(path = path_14x04,
                              sheet = "Freezing",
                              na = "")

## import 14x04 testing data
test_14x04 <- read_excel(path = path_14x04,
                              sheet = "Test Day",
                              na = "")
```

```r
## import the context_order for the training days
context_order <- read_excel(path = path_14x04,
                              sheet = "context_order",
                              na = "")
```

Import behavioral group assignments

```r
## import treatment group assignments
grp_assigns <- read_excel(path = path_grps,
                              sheet = "group_assignment",
                              na = "")
```

Initial Inspection of grp_assigns

Overview of grp_assigns

```r
## use glimpse() from the tibble package (tidyverse)
glimpse(grp_assigns)
```

```
## Observations: 48
## Variables: 2
## $ group <chr> "SAL-LPS", "MIN-SAL", "SAL-SAL", "MIN-LPS", "SAL-LPS", "...
## $ rat   <chr> "14x02_01", "14x02_02", "14x02_03", "14x02_04", "14x02_0...
```

The `grp_assigns` looks good and does not need any further processing.

Initial Training Data Inspection:

Here we make an initial inspection of the imported behavioral training datasets and identify issues to resolve in order to generate the corresponding tidy datasets.

14x02 Training Data Inspection

Overview of the train_14x02 data:

```r
## overview of train_14x02
glimpse(train_14x02)
```

```
## Observations: 311
## Variables: 20
## $ Subject                <chr> "14x02_01", "14x02_02", "14x02_03", "14...
## $ Day of Training        <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ S+ Fr Min 1 sec        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr Min 1            <dbl> 0.00000, 0.00000, 0.00000, 0.00000, 0.0...
## $ S+ Fr Min 2 sec        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr Min 2            <dbl> 0.00000, 0.00000, 0.00000, 0.00000, 0.0...
## $ S+ Fr Min 3 sec        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr Min 3            <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr 20s (sec)        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr min1-3           <dbl> 0.00000, 0.00000, 0.00000, 0.00000, 0.0...
## $ S-FrMin1 Sec           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 1            <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 2 sec        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 2            <dbl> 0.00000, 0.00000, 0.00000, 0.00000, 0.0...
## $ S- Fr Min 3 sec        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 3            <dbl> 0.00000, 0.00000, 0.00000, 0.00000, 0.0...
## $ S- Fr 20s              <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 1-3          <dbl> 0.00000, 0.00000, 0.00000, 0.00000, 0.0...
## $ Diff Fr Min1&2 S+ - S- <dbl> 0.00000, 0.00000, 0.00000, 0.00000, 0.0...
## $ Disc Ratio Fr          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
```

From this view of the 14x02 training data we see that:

- There are 311 rows and 20 variables
- The data appear to be in a tidy format
- Rat identifiers are stored in the "Subject" variable
- The raw freezing data of interest is in numeric variables with names containing the phrases "Min 1 sec", ".Min 2 sec", and "Min 3 sec"
- The remaining numeric variables are values calculated from the raw data
- Freezing data for context A is stored in variables with names starting with "S+"
- Freezing data for context A' is stored in variables with names starting with "S-"

Inspection of missing values:

Are there rows in the 14x02 training data set that do not have **any** raw freezing data? If so, how many rows? To answer this question we look for rows in which the value for *all* the variables containing the raw freezing data are *NA* (missing).

First, we can look at a vector of the number of missing raw data values for each observation.

```
rowSums(is.na(train_14x02[,c(3,5,7,11,13,15)]))
```

```
##   [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [36] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [71] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [106] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [141] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 5 0 6 0 0 0 0 0 0 0 6 0 0 0 0 0 0 6 0 0
## [176] 0 0 0 0 6 0 6 6 0 0 0 0 0 6 6 0 0 0 6 0 0 6 0 0 0 0 0 0 6 0 6 6 0 0 6
## [211] 0 0 6 6 0 0 6 6 0 0 6 0 0 0 0 0 6 6 0 6 6 0 0 6 0 0 6 6 6 0 6 6 6 0 6
## [246] 0 6 0 6 6 6 6 0 6 6 0 0 6 0 0 6 6 6 0 6 6 6 0 6 0 6 0 6 6 6 6 0 6 6 0
## [281] 0 6 0 0 6 6 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

From this vector, we can see that starting with row (observation) 156, there are several observations missing 5 or 6 of the 6 values. We can now find how many observations are missing >=5 values.

```
## use select(), filter() and summarise() from dplyr

raw_dat <- train_14x02 %>%
    select(3,5,7,11,13,15)

 num_NA_rows_14x02 <- raw_dat %>%
    filter(rowSums(is.na(raw_dat)) >= 5) %>%
    summarise(num_rows = n())
```

There are 58 rows (observations) without any raw freezing data, containing only NA values. These observations will be removed during the Data Cleaning step.

14x04 Training Data Inspection

Overview of the train_14x04 data:

```r
## overview of train_14x04
glimpse(train_14x04)
```

```
## Observations: 233
## Variables: 20
## $ Subject               <chr> "14x04_01", "14x04_02", "14x04_03", "14...
## $ Day of Training       <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...
## $ S+ Fr Min 1 sec       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr Min 1           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr Min 2 sec       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr Min 2           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr Min 3 sec       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr Min 3           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr 20s (sec)       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S+ Fr min1-3          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S-FrMin1 Sec          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 1           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 2 sec       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 2           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 3 sec       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 3           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr 20s             <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ S- Fr Min 1-3         <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ Diff Fr Min1&2 S+ - S- <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ Disc Ratio Fr         <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
```

From this view of the 14x04 training data we see that:

- There are 233 rows and 20 variables
- The data appear to be in exactly the same format as the 14x02 training data, with the same variables containing the same types of values

Inspection of missing values:

Similar to our inspection of the 14x02 training data we are interested in knowing how many rows (if any) do not have any raw freezing data. Again, we look for rows in which the value for all the variables containing the raw freezing data are "NA".

First, we can look at a vector of the number of missing raw data values for each observation.

```r
rowSums(is.na(train_14x04[,c(3,5,7,11,13,15)]))
```

```
##   [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [36] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [71] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 6 0 0 0 0 0 0 0
## [106] 0 0 0 0 0 0 0 0 0 0 6 0 0 0 0 0 6 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 6
## [141] 0 0 0 0 0 6 0 6 6 0 6 0 0 0 6 0 6 6 0 0 0 6 6 6 0 0 6 0 0 6 0 6 6 0 6
## [176] 6 0 0 6 0 6 6 0 0 0 6 6 6 6 6 6 0 0 6 6 6 6 6 6 6 6 6 6 6 6 6 0 0 0 6
## [211] 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 0 6
```

From this vector, we can see that starting with row 98, there are several observations missing 6 of the 6 values. We can now find how many observations are missing 6 values.

```
## use select(), filter() and summarise() from dplyr

raw_dat <- train_14x04 %>%
    select(3,5,7,11,13,15)

 num_NA_rows_14x04 <- raw_dat %>%
    filter(rowSums(is.na(raw_dat)) == 6) %>%
    summarise(num_rows = n())

 rm(raw_dat)
```

There are 65 observations without any raw freezing data. These observations will be removed during the Data Cleaning step.

Training Data Cleaning

In this step the identified issues in the 14x02 and 14x04 behavioral training data are addressed.

Select variables containing raw training data

The dataset contains variables that contain values calculated from the raw data, rather than the raw data itself. These variables will be removed and later re-calculated during the Data Transformation step in order to maintain transparent and reproducible data analysis.

```
## use the select() function of dplyr (tidyverse)

## select 14x02 vars
train_14x02 <- train_14x02 %>%
    select(1:3,5,7,11,13,15)

## select 14x04 vars
train_14x04 <- train_14x04 %>%
    select(1:3,5,7,11,13,15)
```

Rename training data variables

Some of the variable names are not in a correct syntax form. All the variable names will be changed to more descriptive and syntactically correct names.

```
## set up a vector with the new variable names
vars <- c("rat", "training_day", "A_min1", "A_min2", "A_min3",
          "A'_min1", "A'_min2", "A'_min3")

## rename 14x02 vars
names(train_14x02) <- vars

## rename 14x04 vars
names(train_14x04) <- vars
```

We can inspect the resulting datasets:

14x02 training data:

```
glimpse(train_14x02)
```

```
## Observations: 311
## Variables: 8
## $ rat          <chr> "14x02_01", "14x02_02", "14x02_03", "14x02_04", "...
## $ training_day <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ A_min1       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ A_min2       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ A_min3       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ A'_min1      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ A'_min2      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ A'_min3      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
```

14x04 training data:

```
glimpse(train_14x04)
```

```
## Observations: 233
## Variables: 8
## $ rat          <chr> "14x04_01", "14x04_02", "14x04_03", "14x04_04", "...
## $ training_day <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
## $ A_min1       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ A_min2       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ A_min3       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ A'_min1      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ A'_min2      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
## $ A'_min3      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
```

Remove observations without training data

The rows (observations) without any freezing data do not add to the dataset and will be removed. For `train_14x02` any row missing 5 or 6 values will be removed, and for `train_14x04` any row missing 6 values will be removed.

```
## remove rows missing data from train_14x02
train_14x02 <- train_14x02 %>%
    filter(!rowSums(is.na(train_14x02[,3:8])) >= 5)

## remove rows missing data from train_14x04
train_14x04 <- train_14x04 %>%
    filter(!rowSums(is.na(train_14x04[,3:8])) == 6)
```

The `train_14x02` dataset now has 253 observations and the `train_14x04` dataset now has 168 observations

Add a ‘study’ variable to each training dataset

Here, we add a new variable for ‘study’ to each training dataset to distinguish the study as 14x02 or 14x04:

```
## use the mutate() function of dplyr

## add the study value of 14x02
train_14x02 <- train_14x02 %>%
    mutate(study = "14x02")

## add the study value of 14x04
```

```
train_14x04 <- train_14x04 %>%
    mutate(study = "14x04")
```

Combine 14x02 and 14x04 training datasets

Next, the rows of the 14x04 training data can be appended to the 14x02 training data to generate a compile `train_dat` data table. The variables are then reordered.

```
## use the bind_rows() function of dplyr
train_dat <- bind_rows(train_14x02, train_14x04)

## reorder the variables
train_dat <- select(train_dat,
                    1, 9, 2:8)
```

Add a 'context_order' variable to the training data

Finally, we add context order information for each training day and reorder the variables:

```
## use the left_join() function from dplyr to add
##   the context_order var and preserve the train_dat data
train_dat <- left_join(train_dat, context_order,
                       by = "training_day")

## reorder the variables
train_dat <- select(train_dat,
                    1:3, 10, 4:9)
```

Add the treatment 'group' variable to the training dataset

One of the experimental objectives is to determine the effect of neuroinflammation on contextual memory. Neuroinflammation was induced by systemic administration of lipopolysaccride (LPS). Secondarily, to determine if LPS caused memory deficits via cytokine release by microglia, some rats were also treated with minocycline (MIN) to block microglia activation. As a control, groups of rats were gives saline in place of either minocycline or LPS. Thus, there were four treatment groups:
1. saline-saline (SAL-SAL); control rats
2. saline-LPS (SAL-LPS); rats given saline, instead of minocycline, followed by LPS
3. minocycline-saline (MIN-saline); rats given minocycline, followed by saline
4. minocycline-LPS (MIN-LPS); rats given minocycline, followed by LPS
5. caged-control (CC); rats that not treated and are not tested in the context discrimination task

Here the treatment group variable is added to the behavioral training data:

```
## use the left_join() function from dplyr to add
##    the group var and preserve the train_dat data
train_dat <- left_join(train_dat, grp_assigns,
                       by = "rat")

## reorder the variables
train_dat <- select(train_dat,
                    1,11, 2:10)
```

Convert variables to factor variables

In the tidy training dataset (`train_dat`) there are 421 observations and 11 variables. To facilitate downstream data analysis, the variables 'study', 'group', 'training_day' and 'context_order' are converted to factor variables:

```
## use the mutate() function of dplyr
train_dat <- train_dat %>%
    mutate(study = as.factor(study),
           group = as.factor(group),
        training_day = as.factor(training_day),
        context_order = as.factor(context_order)
    )
```

Overview of final form of the behavioral training data:

```
glimpse(train_dat)
```

```
## Observations: 421
## Variables: 11
## $ rat           <chr> "14x02_01", "14x02_02", "14x02_03", "14x02_04", ...
## $ group         <fctr> SAL-LPS, MIN-SAL, SAL-SAL, MIN-LPS, SAL-LPS, SA...
## $ study         <fctr> 14x02, 14x02, 14x02, 14x02, 14x02, 14x02, 14x02...
## $ training_day  <fctr> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ context_order <fctr> A'A, A'A, A'A, A'A, A'A, A'A, A'A, A'A, A'A, A'...
## $ A_min1        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ A_min2        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ A_min3        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ A'_min1       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ A'_min2       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
## $ A'_min3       <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
```

A sample of 15 observations from the `train_dat` dataset:

```
## view random rows of the train_dat table
set.seed(563)

pander(head(sample_n(train_dat, 421), 15),
       caption = "Sample of the training data",
       caption.placement = "top",
       digits = 3, style = 'rmarkdown', split.table = "Inf")
```

Table 1: Sample of the training data

rat	group	study	training_day	context_order	A_min1	A_min2	A_min3	A'_min1	A'_min2	A'_min3
14x04_24	MIN-SAL	14x04	1	A'A	0	0	0	0	0	0
14x02_10	MIN-LPS	14x02	3	AA'	0	3	23	0	2	26
14x02_12	SAL-LPS	14x02	2	AA'	0	4	0	14	13	33
14x04_21	MIN-SAL	14x04	6	AA'	27	36	32	15	0	0
14x02_12	SAL-LPS	14x02	6	AA'	9	9	32	0	0	4
14x04_07	CC	14x04	5	A'A	2	8	5	3	0	2

rat	group	study	training_day	context_order	A_min1	A_min2	A_min3	A'_min1	A'_min2	A'_min3
14x02_19	CC	14x02	13	AA'	0	2	0	0	0	0
14x02_21	MIN-LPS	14x02	4	A'A	0	10	2	0	0	0
14x02_24	MIN-LPS	14x02	7	A'A	0	0	33	0	3	0
14x04_01	SAL-SAL	14x04	3	AA'	0	0	14	0	0	0
14x02_20	MIN-SAL	14x02	4	A'A	0	6	0	0	4	0
14x02_14	MIN-SAL	14x02	4	A'A	0	16	25	3	0	0
14x02_24	MIN-LPS	14x02	6	AA'	0	4	3	0	0	0
14x04_12	SAL-SAL	14x04	5	A'A	6	7	30	0	7	28
14x04_24	MIN-SAL	14x04	2	AA'	0	0	0	5	7	5

Initial Testing Data Inspection

Here we make an initial inspection of the imported behavioral testing datasets and identify issues to resolve in order to generate the corresponding tidy datasets.

14x02 Testing Data Inspection

Overview of the test_14x02 data:

```
## overview of test_14x02
glimpse(test_14x02)
```

```
## Observations: 21
## Variables: 23
## $ Subject                  <chr> "14x02_12", "14x02_14", "14x02_22", "14...
## $ S+ Fr Min 1 sec          <dbl> 14, 10, 31, 0, 4, 24, 8, 2, 42, 4, 9, 1...
## $ S+ Fr Min 2 sec          <dbl> 43, 23, 54, 0, 36, 49, 21, 22, 55, 30, ...
## $ S+ Fr Min 3 sec          <dbl> 38, 41, 43, 3, 56, 45, 27, 40, 52, 52, ...
## $ S+ Fr Min4 sec           <dbl> 57, 30, 49, 9, 57, 46, 42, 15, 48, 43, ...
## $ S+ Fr Min 5 sec          <dbl> 36, 17, 40, 0, 47, 7, 26, 18, 56, 22, 3...
## $ S+ Fr Min 1              <dbl> 23.333333, 16.666667, 51.666667, 0.0000...
## $ S+ Fr Min 2              <dbl> 71.66667, 38.33333, 90.00000, 0.00000, ...
## $ Min 3                    <dbl> 63.33333, 68.33333, 71.66667, 5.00000, ...
## $ Min 4                    <dbl> 95.00000, 50.00000, 81.66667, 15.00000,...
## $ Min 5                    <dbl> 60.00000, 28.33333, 66.66667, 0.00000, ...
## $ S+ Fr min1-3             <dbl> 52.777778, 41.111111, 71.111111, 1.6666...
## $ TOTAL % FR A             <dbl> 62.666667, 40.333333, 72.333333, 4.0000...
## $ S-FrMin1 Sec             <dbl> 0, 16, 2, 0, 0, 0, 5, 6, 18, 0, 3, 0, 0...
## $ S- Fr Min 2 sec          <dbl> 3, 48, 40, 1, 0, 0, 0, 0, 24, 0, 0, 2, ...
## $ S- Fr Min 3 sec          <dbl> 2, 52, 21, 0, 0, 4, 2, 0, 36, 0, 0, 0, ...
## $ S- Fr 20s                <dbl> 0, 0, 17, 0, 0, 10, 0, 0, 8, 0, 0, 0, 0...
## $ Min 1                <dbl> 0.000000, 26.666667, 3.333333, 0.000000...
## $ Min 2                <dbl> 5.000000, 80.000000, 66.666667, 1.66666...
## $ Min 3                <dbl> 3.333333, 86.666667, 35.000000, 0.00000...
## $ S- Fr Min 1&3        <dbl> 2.7777778, 64.4444444, 35.0000000, 0.55...
## $ Diff Fr Min1&2 S+ - S- <dbl> 50.0000000, -23.3333333, 36.1111111, 1....
## $ Disc Ratio Fr        <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
```

From this view of the 14x02 testing data we see that:

- There are 21 rows and 23 variables
- The data appear to be in a tidy format
- Rat identifiers are stored in the "Subject" variable
- Rat identifiers "14x02_1", "14x02_2", "14x02_3", "14x02_5", "14x02_7", "14x02_9", should be in the form of "14x02_0#"
- The raw freezing data of interest is in numeric variables with names containing the phrases "Min n sec". There are 5 columns with raw context A freezing data and 3 columns with raw context A' freezing data
- The remaining numeric variables are values calculated from the raw data
- Freezing data for context A is stored in variables with names starting with "S+"
- Freezing data for context A' is stored in variables with names starting with "S-"

Inspection of missing values:

We will use the same procedure here as we used for the training data to determine if- and how many rows do not contain raw freezing data.

First, we can look at a vector of the number of missing raw data values for each observation.

```
rowSums(is.na(test_14x02[,c(2:6, 14:16)]))
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
```

From this vector, we can see that observation 18 has 1 missing value. We can take a closer look at this row.

```
pander(test_14x02[18,c(1:6, 14:16)],
       style = 'rmarkdown', split.table = "Inf")
```

Subject	S+ Fr Min 1 sec	S+ Fr Min 2 sec	S+ Fr Min 3 sec	S+ Fr Min4 sec	S+ Fr Min 5 sec	S- FrMin1 Sec	S- Fr Min 2 sec	S- Fr Min 3 sec
14x02_04	15	11	25	17	57	0	NA	0

We can see that the only value missing is that for minute 2 of context A'. Since, the other values for context A' are 0, we will assume that the value of minute 2 is also 0 and will set the value to 0.

```
## change the NA in row 18, column 15 to 0
test_14x02[18,15] <- 0
```

Another look:

```
rowSums(is.na(test_14x02[,c(2:6, 14:16)]))
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Thus, the test_14x02 dataset does not have any missing values for the raw freezing data.

14x04 Testing Data Inspection

Overview of the `test_14x04` dataset:

```
## overview of test_14x04
glimpse(test_14x04)
```

```
## Observations: 42
## Variables: 23
## $ Subject               <chr> "14x02_04", "14x02_06", "14x02_08", "14...
## $ S+ Fr Min 1 sec       <dbl> 15, 18, 0, 42, 26, 4, 14, 28, 10, 4, 2,...
## $ S+ Fr Min 2 sec       <dbl> 11, 32, 6, 55, 28, 30, 43, 21, 23, 36, ...
## $ S+ Fr Min 3 sec       <dbl> 25, 40, 0, 52, 32, 52, 38, 30, 41, 56, ...
## $ S+ Fr Min4 sec        <dbl> 17, 44, 6, 48, 35, 43, 57, 14, 30, 57, ...
## $ S+ Fr Min 5 sec       <dbl> 57, 33, 10, 56, 50, 22, 36, 19, 17, 47,...
## $ S+ Fr Min 1           <dbl> 25.000000, 30.000000, 0.000000, 70.0000...
## $ S+ Fr Min 2           <dbl> 18.33333, 53.33333, 10.00000, 91.66667,...
## $ Min 3                 <dbl> 41.66667, 66.66667, 0.00000, 86.66667, ...
## $ Min 4                 <dbl> 28.33333, 73.33333, 10.00000, 80.00000,...
## $ Min 5                 <dbl> 95.00000, 55.00000, 16.66667, 93.33333,...
## $ S+ Fr min1-3          <dbl> 28.333333, 50.000000, 3.333333, 82.7777...
## $ TOTAL % FR A          <dbl> 41.666667, 55.666667, 7.333333, 84.3333...
## $ S-FrMin1 Sec          <dbl> 0, 20, 0, 18, 18, 0, 0, 5, 16, 0, 6, 5,...
## $ S- Fr Min 2 sec       <dbl> NA, 10, 2, 24, 7, 0, 3, 4, 48, 0, 0, 0,...
## $ S- Fr Min 3 sec       <dbl> 0, 43, 5, 36, 24, 0, 2, 22, 52, 0, 0, 2...
## $ S- Fr 20s             <dbl> 0, 16, 0, 8, 20, 0, 0, 10, 0, 0, 0, 0, ...
## $ Min 1                 <dbl> 0.000000, 33.333333, 0.000000, 30.00000...
## $ Min 2                 <dbl> 0.000000, 16.666667, 3.333333, 40.00000...
## $ Min 3                 <dbl> 0.000000, 71.666667, 8.333333, 60.00000...
## $ S- Fr Min 1&3         <dbl> 0.0000000, 40.5555556, 3.8888889, 43.33...
## $ Diff Fr Min1&2 S+ - S- <dbl> 28.3333333, 9.4444444, -0.5555556, 39.4...
## $ Disc Ratio Fr         <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,...
```

From this view of the 14x04 testing data we can see that:

- There are 42 rows and 23 variables
- The data appears to be in exactly the same format as the `test_14x02` dataset
- The variables are exactly the same as in the `test_14x02` dataset
- The dataset contains rats from both 14x04 **and** 14x02 (which will be removed)

Inspection of missing values:

We will use the same procedure here as we used for the `test_14x02` dataset to determine if- and how many rows do not contain raw freezing data.

We can look at a vector of the number of missing raw data values for each observation.

```
rowSums(is.na(test_14x04[,c(2:6, 14:16)]))
```

```
##  [1] 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [36] 0 0 0 0 0 0 0
```

Row 1 of the `test_14x04` dataset has 1 missing value. However, from the overview of the dataset, we can see that the first row is for a rat in the 14x02 study. This row will be removed during the data cleaning step and we don't need to do anything here.

Testing data Data Cleaning

In this step the identified issues in the 14x02 and 14x04 behavioral testing data are addressed.

Select variables containing raw testing data

The dataset contains variables that contain values calculated from the raw data, rather than the raw data itself. These variables will be removed and later re-calculated during the Data Transformation step in order to maintain transparent and reproducible data analysis.

```
## use base subsetting to get around problem of duplicate var name
## select 14x02 vars
test_14x02 <- test_14x02[, c(1:6,14:16)]

## select 14x04 vars
test_14x04 <- test_14x04[, c(1:6,14:16)]
```

Rename testing data variables

Some of the variable names are not in a correct syntax form. All the variable names will be changed to more descriptive and syntactically correct names.

```
## set up a vectors with the new variable names
vars <- c("rat", "A_min1", "A_min2", "A_min3",
          "A_min4", "A_min5",
          "A'_min1", "A'_min2", "A'_min3")


## rename 14x02 vars
names(test_14x02) <- vars

## rename 14x04 vars
names(test_14x04) <- vars
```

Correct inconsistent rat identifiers in 14x02 testing data

Rat identifiers "14x02_1", "14x02_2", "14x02_3", "14x02_5", "14x02_7", "14x02_9", should be in the form of "14x02_0#".

```
test_14x02$rat <- ifelse(test_14x02$rat == "14x02_1", "14x02_01",
                    ifelse(test_14x02$rat == "14x02_2", "14x02_02",
                    ifelse(test_14x02$rat == "14x02_3", "14x02_03",
                    ifelse(test_14x02$rat == "14x02_5", "14x02_05",
                    ifelse(test_14x02$rat == "14x02_7", "14x02_07",
                    ifelse(test_14x02$rat == "14x02_9", "14x02_09",
                           test_14x02$rat
                           ))))))
```

We can inspect the resulting datasets:
14x02 testing data:

```
glimpse(test_14x02)
```

```
## Observations: 21
## Variables: 9
## $ rat     <chr> "14x02_12", "14x02_14", "14x02_22", "14x02_05", "14x02...
## $ A_min1  <dbl> 14, 10, 31, 0, 4, 24, 8, 2, 42, 4, 9, 13, 8, 20, 26, 1...
## $ A_min2  <dbl> 43, 23, 54, 0, 36, 49, 21, 22, 55, 30, 23, 29, 31, 35,...
## $ A_min3  <dbl> 38, 41, 43, 3, 56, 45, 27, 40, 52, 52, 25, 46, 46, 32,...
## $ A_min4  <dbl> 57, 30, 49, 9, 57, 46, 42, 15, 48, 43, 17, 46, 52, 29,...
## $ A_min5  <dbl> 36, 17, 40, 0, 47, 7, 26, 18, 56, 22, 37, 38, 36, 39, ...
## $ A'_min1 <dbl> 0, 16, 2, 0, 0, 0, 5, 6, 18, 0, 3, 0, 0, 14, 18, 20, 5...
## $ A'_min2 <dbl> 3, 48, 40, 1, 0, 0, 0, 0, 24, 0, 0, 2, 4, 30, 7, 10, 4...
## $ A'_min3 <dbl> 2, 52, 21, 0, 0, 4, 2, 0, 36, 0, 0, 0, 2, 45, 24, 43, ...
```

14x04 testing data:

```
glimpse(test_14x04)
```

```
## Observations: 42
## Variables: 9
## $ rat     <chr> "14x02_04", "14x02_06", "14x02_08", "14x02_1", "14x02_...
## $ A_min1  <dbl> 15, 18, 0, 42, 26, 4, 14, 28, 10, 4, 2, 8, 11, 24, 31,...
## $ A_min2  <dbl> 11, 32, 6, 55, 28, 30, 43, 21, 23, 36, 22, 21, 21, 49,...
## $ A_min3  <dbl> 25, 40, 0, 52, 32, 52, 38, 30, 41, 56, 40, 27, 27, 45,...
## $ A_min4  <dbl> 17, 44, 6, 48, 35, 43, 57, 14, 30, 57, 15, 42, 33, 46,...
## $ A_min5  <dbl> 57, 33, 10, 56, 50, 22, 36, 19, 17, 47, 18, 26, 28, 7,...
## $ A'_min1 <dbl> 0, 20, 0, 18, 18, 0, 0, 5, 16, 0, 6, 5, 0, 0, 2, 3, 0,...
## $ A'_min2 <dbl> NA, 10, 2, 24, 7, 0, 3, 4, 48, 0, 0, 0, 0, 0, 40, 0, 0...
## $ A'_min3 <dbl> 0, 43, 5, 36, 24, 0, 2, 22, 52, 0, 0, 2, 0, 4, 21, 0, ...
```

Remove 14x02 observations from the 14x04 testing dataset

During inspection of the 14x04 testing data it was found that the dataset also contained some 14x02 subjects. These rats are removed from the 14x04 dataset here.

```
## use the str_detect() function from the stringr package

## filter test_14x04 for rat ids beginning with "14x04"
test_14x04 <- test_14x04 %>%
    filter(str_detect(rat, "14x04"))
```

Add the 'study' variable to testing data

Here, we add a new variable for 'study' to each testing dataset to distinguish the study as 14x02 or 14x04:

```
## use the mutate() function of dplyr

## add the study value of 14x02
test_14x02 <- test_14x02 %>%
    mutate(study = "14x02")

## add the study value of 14x04
test_14x04 <- test_14x04 %>%
    mutate(study = "14x04")
```

Combine 14x02 and 14x04 testing datasets

Next, the rows of the 14x04 testing data are appended to the 14x02 testing data to generate a compile `test_dat` data table. The variables are then reordered.

```
## use the bind_rows() function of dplyr
test_dat <- bind_rows(test_14x02, test_14x04)

## reorder the variables
test_dat <- select(test_dat,
                   1, 10, 2:9)
```

Add the treatment 'group' variable to the testing dataset

Here the treatment group variable is added to the behavioral testing data:

```
## use the left_join() function from dplyr to
##    add the group var and preserve the test_dat data
test_dat <- left_join(test_dat, grp_assigns, by = "rat")

## reorder the variables
test_dat <- select(test_dat,
                   1,11, 2:10)
```

Convert categorical variables to factor variables

In the tidy testing dataset (`test_dat`) there are 42 observations and 11 variables. To facilitate downstream data analysis, the categorical variables 'study', and 'group' are converted to factor variables:

```
## use the mutate() function of dplyr
test_dat <- test_dat %>%
    mutate(study = as.factor(study),
           group = as.factor(group))
```

Overview of final form of the behavioral testing data:

```
glimpse(test_dat)
```

```
## Observations: 42
## Variables: 11
## $ rat     <chr> "14x02_12", "14x02_14", "14x02_22", "14x02_05", "14x02...
## $ group   <fctr> SAL-LPS, MIN-SAL, MIN-LPS, SAL-LPS, SAL-SAL, MIN-LPS,...
## $ study   <fctr> 14x02, 14x02, 14x02, 14x02, 14x02, 14x02, 14x02, 14x0...
## $ A_min1  <dbl> 14, 10, 31, 0, 4, 24, 8, 2, 42, 4, 9, 13, 8, 20, 26, 1...
## $ A_min2  <dbl> 43, 23, 54, 0, 36, 49, 21, 22, 55, 30, 23, 29, 31, 35,...
## $ A_min3  <dbl> 38, 41, 43, 3, 56, 45, 27, 40, 52, 52, 25, 46, 46, 32,...
## $ A_min4  <dbl> 57, 30, 49, 9, 57, 46, 42, 15, 48, 43, 17, 46, 52, 29,...
## $ A_min5  <dbl> 36, 17, 40, 0, 47, 7, 26, 18, 56, 22, 37, 38, 36, 39, ...
## $ A'_min1 <dbl> 0, 16, 2, 0, 0, 0, 5, 6, 18, 0, 3, 0, 0, 14, 18, 20, 5...
## $ A'_min2 <dbl> 3, 48, 40, 1, 0, 0, 0, 0, 24, 0, 0, 2, 4, 30, 7, 10, 4...
## $ A'_min3 <dbl> 2, 52, 21, 0, 0, 4, 2, 0, 36, 0, 0, 0, 2, 45, 24, 43, ...
```

A sample of 15 observations from the `test_dat` dataset:

```r
## view random rows of the test_dat table
set.seed(543)

pander(head(sample_n(test_dat, 42), 15),
       caption = "Sample of the testing data",
       caption.placement = "top",
       digits = 3, style = 'rmarkdown', split.table = "Inf")
```

Table 3: Sample of the testing data

rat	group	study	A_min1	A_min2	A_min3	A_min4	A_min5	A'_min1	A'_min2	A'_min3
14x04_21	MIN-SAL	14x04	16	28	44	46	46	0	0	0
14x04_15	SAL-LPS	14x04	31	50	40	45	46	46	42	0
14x04_02	MIN-SAL	14x04	47	58	60	60	60	5	12	21
14x02_15	SAL-SAL	14x02	4	36	56	57	47	0	0	0
14x04_05	SAL-SAL	14x04	0	16	47	18	25	5	0	0
14x04_14	MIN-SAL	14x04	6	21	39	38	34	0	2	0
14x02_06	SAL-LPS	14x02	18	32	40	44	33	20	10	43
14x02_18	SAL-SAL	14x02	2	22	40	15	18	6	0	0
14x02_21	MIN-LPS	14x02	24	49	45	46	7	0	0	4
14x02_11	SAL-SAL	14x02	4	30	52	43	22	0	0	0
14x04_04	SAL-LPS	14x04	46	50	51	47	54	19	20	56
14x02_09	SAL-LPS	14x02	20	35	32	29	39	14	30	45
14x02_07	MIN-SAL	14x02	8	31	46	52	36	0	4	2
14x04_18	MIN-LPS	14x04	34	51	39	54	49	29	31	54
14x04_12	SAL-SAL	14x04	0	13	40	34	29	2	0	18

R session information:

```
## Session info -------------------------------------------------------------

##  setting  value
##  version  R version 3.3.3 (2017-03-06)
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  English_United States.1252
##  tz       America/Los_Angeles
##  date     2017-04-03

## Packages -----------------------------------------------------------------

##  package      * version  date       source
##  assertthat     0.1      2013-12-06 CRAN (R 3.3.1)
##  backports      1.0.5    2017-01-18 CRAN (R 3.3.2)
##  base64enc      0.1-3    2015-07-28 CRAN (R 3.3.1)
##  BH             1.62.0-1 2016-11-19 CRAN (R 3.3.2)
##  bitops         1.0-6    2013-08-17 CRAN (R 3.3.1)
##  broom        * 0.4.2    2017-02-13 CRAN (R 3.3.2)
##  caTools        1.17.1   2014-09-10 CRAN (R 3.3.1)
##  colorspace     1.3-2    2016-12-14 CRAN (R 3.3.2)
##  curl           2.4      2017-03-24 CRAN (R 3.3.3)
##  DBI            0.6-1    2017-04-01 CRAN (R 3.3.3)
## dichromat      2.0-0    2013-01-24 CRAN (R 3.3.1)
## digest         0.6.12   2017-01-27 CRAN (R 3.3.2)
## dplyr        * 0.5.0    2016-06-24 CRAN (R 3.3.1)
## evaluate       0.10     2016-10-11 CRAN (R 3.3.1)
## forcats        0.2.0    2017-01-23 CRAN (R 3.3.2)
## foreign        0.8-67   2016-09-13 CRAN (R 3.3.1)
## ggplot2      * 2.2.1    2016-12-30 CRAN (R 3.3.2)
## gtable         0.2.0    2016-02-26 CRAN (R 3.3.1)
## haven          1.0.0    2016-09-23 CRAN (R 3.3.2)
## highr          0.6      2016-05-09 CRAN (R 3.3.1)
## hms            0.3      2016-11-22 CRAN (R 3.3.2)
## htmltools      0.3.5    2016-03-21 CRAN (R 3.3.1)
## httr           1.2.1    2016-07-03 CRAN (R 3.3.1)
## jsonlite       1.3      2017-02-28 CRAN (R 3.3.2)
## knitr        * 1.15.1   2016-11-22 CRAN (R 3.3.2)
## labeling       0.3      2014-08-23 CRAN (R 3.3.1)
## lattice        0.20-35  2017-03-25 CRAN (R 3.3.3)
## lazyeval       0.2.0    2016-06-12 CRAN (R 3.3.1)
## lubridate      1.6.0    2016-09-13 CRAN (R 3.3.1)
## magrittr       1.5      2014-11-22 CRAN (R 3.3.1)
## markdown       0.7.7    2015-04-22 CRAN (R 3.3.1)
## MASS           7.3-45   2016-04-21 CRAN (R 3.3.3)
## mime           0.5      2016-07-07 CRAN (R 3.3.1)
## mnormt         1.5-5    2016-10-15 CRAN (R 3.3.1)
## modelr         0.1.0    2016-08-31 CRAN (R 3.3.2)
## munsell        0.4.3    2016-02-13 CRAN (R 3.3.1)
## nlme           3.1-131  2017-02-06 CRAN (R 3.3.2)
## openssl        0.9.6    2016-12-31 CRAN (R 3.3.2)
## pander       * 0.6.0    2017-03-07 Github (Rapporter/pander@b188a19)
## plyr           1.8.4    2016-06-08 CRAN (R 3.3.1)
## psych          1.7.3.21 2017-03-22 CRAN (R 3.3.3)
## purrr        * 0.2.2    2016-06-18 CRAN (R 3.3.2)
## R6             2.2.0    2016-10-05 CRAN (R 3.3.1)
## RColorBrewer   1.1-2    2014-12-07 CRAN (R 3.3.1)
## Rcpp           0.12.10  2017-03-19 CRAN (R 3.3.3)
## readr        * 1.1.0    2017-03-22 CRAN (R 3.3.3)
## readxl       * 0.1.1    2016-03-28 CRAN (R 3.3.2)
## reshape2       1.4.2    2016-10-22 CRAN (R 3.3.1)
## rmarkdown    * 1.4      2017-03-24 CRAN (R 3.3.3)
## rprojroot      1.2      2017-01-16 CRAN (R 3.3.2)
## rvest          0.3.2    2016-06-17 CRAN (R 3.3.1)
## scales         0.4.1    2016-11-09 CRAN (R 3.3.2)
## selectr        0.3-1    2016-12-19 CRAN (R 3.3.2)
## stringi        1.1.3    2017-03-21 CRAN (R 3.3.3)
## stringr      * 1.2.0    2017-02-18 CRAN (R 3.3.2)
## tibble       * 1.3.0    2017-04-01 CRAN (R 3.3.3)
## tidyr        * 0.6.1    2017-01-10 CRAN (R 3.3.2)
## tidyverse    * 1.1.1    2017-01-27 CRAN (R 3.3.2)
## xml2           1.1.1    2017-01-24 CRAN (R 3.3.2)
## yaml           2.1.14   2016-11-12 CRAN (R 3.3.2)
```