Introduction to R and RStudio

MM4DBER Training Team

Updated: August 22, 2023

MM4DBER Project

Mixture Modeling for Discipline Based Education Researchers (MM4DBER) is an NSF funded training grant to support STEM Education scholars in integrating mixture modeling into their research.

- Please visit our website to learn more and apply for the year-long fellowship.
- Follow us on Twitter!

Visit our GitHub account to download the materials needed for this walkthrough.

Introduction to R and RStudio

This walkthrough is presented by the MM4DBER team and will go through some common tasks carried out in R. There are many free resources available to get started with R and RStudio. One of our favorites is *R for Data Science*.

PART 1: Installation

Install: R, RStudio, and Mplus

- **Install R/Rstudio**: Here you will find a guide to installing both R and R Studio.
- **Install Mplus**: To install the Mplus software go to their website here.

Note: The installation of Mplus requires a paid license with the mixture add-on. MM4DBER fellows will be given their own copy of Mplus for use during the one year training.

PART 2: Set-up

Step 1: Create a new R-project in RStudio

R-projects help us organize our folders , filepaths, and scripts. To create a new R project:

- File –> New Project...

Click "New Directory" –> New Project –> Name your project

Step 2: Create an R-markdown document

An R-markdown file provides an authoring framework for data science that allows us to organize our reports using texts and code chunks. This document you are reading was made using R-markdown!

To create an R-markdown:

- File –> New File –> R Markdown...

In the window that pops up, give the R-markdown a title such as "**Introduction to R and RStudio**" Click "OK." You should see a new markdown with some example text and code chunks. We want a clean document to start off with so delete everything from line 10 down. Go ahead and save this document in your R Project folder.

Step 3: Load packages

Your first code chunk in any given markdown should be the packages you will be using. To insert a code chunk, etiher use the keyboard shortcut ctrl + alt + i or Code –> Insert Chunk or click the green box with the letter C on it. There are a few packages we want our markdown to read in:

```
library(psych)     # describe()
library(here)      # helps with specifying file paths
library(gt)        # create tables
library(tidyverse) # collection of R packages designed for data science
```

As a reminder, if a function does not work and you receive an error like this: `could not find function "random_function"`; or if you try to load a package and you receive an error like this: `there is no package called `random_package`` , then you will need to install the package using `install.packages("random_package")` in the console (the bottom-left window in R studio).

Once you have installed the package you will *never* need to install it again, however you must *always* load in the packages at the beginning of your R markdown using `library(random_package)`, as shown in this document.

The style of code and package we will be using is called `tidyverse`. Most functions we use for data manipulation are available within the `tidyverse` package and if not, I've indicated the packages used in the code chunk above.

PART 3: Explore the data

Step 4: Read in data

To demonstrate mixture modeling in the training program of the NSF grant we utilize the *Longitudinal Study of American Youth (LSAY)* data repository.

Table 1:

LSAY Variable Descriptions.

Name	Label	Values
Enjoy	I enjoy science	0 = Disagree, 1 = Agree
Useful	Science useful in everday problems	0 = Disagree, 1 = Agree
Logical	Science helps logical thinkng	0 = Disagree, 1 = Agree
Job	Need science for a good job	0 = Disagree, 1 = Agree
Adult	Will use science often as an adult	0 = Disagree, 1 = Agree
Female	Reported gender	0 = Male, 1 = Female

To read in data in R:

```
data <- read_csv(here("data", "lsay_sci_data.csv"))
```

View data in R:

```
# 1. click on the data in your Global Environment (upper right pane) or use...
View(data)
# 2. summary() gives basic summary statistics & shows number of NA values
# *great for checking that data has been read in correctly*
summary(data)
```

```
##      Enjoy            Useful          Logical            Job
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000   Median :0.0000   Median :0.0000   Median :0.0000
##  Mean   :0.6131   Mean   :0.4036   Mean   :0.4923   Mean   :0.4037
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##  NA's   :19       NA's   :73       NA's   :69       NA's   :49
##      Adult            Female
##  Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.0000   Median :0.0000
##  Mean   :0.4614   Mean   :0.4812
##  3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000
##  NA's   :18
```

```
# 3. names() provides a list of column names. Very useful if you don't have them memorized!
names(data)
```

```
## [1] "Enjoy"   "Useful"  "Logical" "Job"     "Adult"   "Female"
```

```
# 4. head() prints the top x rows of the dataframe
head(data)
```

```
## # A tibble: 6 x 6
##   Enjoy Useful Logical   Job Adult Female
##   <dbl>  <dbl>   <dbl> <dbl> <dbl>  <dbl>
## 1     1      1       1     1     1      0
## 2     0      0       1     0     0      1
## 3     1      1       0     0     0      0
## 4     0      0       0     1     1      0
## 5     0      1       1     0     0      0
## 6     0      0       0     0     0      1
```

Step 5: Select Columns and Filter Rows

```
# Select columns one at a time
data_attitudes <- data %>%
  select(Enjoy, Useful, Logical)

# Select columns left to right
data_attitudes <- data %>%
  select(Enjoy:Adult)

# Remove columns
data_attitudes <- data %>%
  select(-Female)
```

What if we want to look at a subset of the data?

- For example, what if we want to subset the data for female science attitudes? (`Female`)
- We can use `tidyverse::filter()` to subset the data using certain criteria.

```
# Filter rows
data_female <- data %>%
  filter(Female == 1)

# You can use any operator to filter: >, <, ==, >=, etc.

data_female %>% nrow()
```

```
## [1] 1473
```

Step 6: Descriptive Statistics

Let's look at descriptive statistics for each of the science attitude variables.

```
data_attitudes %>%
  summary()
```

```
##      Enjoy            Useful          Logical            Job
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000   Median :0.0000   Median :0.0000   Median :0.0000
##  Mean   :0.6131   Mean   :0.4036   Mean   :0.4923   Mean   :0.4037
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##  NA's   :19       NA's   :73       NA's   :69       NA's   :49
##      Adult
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.4614
##  3rd Qu.:1.0000
##  Max.   :1.0000
##  NA's   :18
```

Alternatively, we can use the `psych::describe()` function to give more information:

```
data_attitudes %>%
  describe()
```

```
##         vars    n mean   sd median trimmed mad min max range  skew kurtosis
## Enjoy      1 3042 0.61 0.49      1    0.64   0   0   1     1 -0.46    -1.79
## Useful     2 2988 0.40 0.49      0    0.38   0   0   1     1  0.39    -1.85
## Logical    3 2992 0.49 0.50      0    0.49   0   0   1     1  0.03    -2.00
## Job        4 3012 0.40 0.49      0    0.38   0   0   1     1  0.39    -1.85
## Adult      5 3043 0.46 0.50      0    0.45   0   0   1     1  0.15    -1.98
##           se
## Enjoy   0.01
## Useful  0.01
```

```
## Logical 0.01
## Job     0.01
## Adult   0.01
```

Since we have binary data, it would be helpful to look at variable proportions:

```
data %>%
  drop_na() %>%
  pivot_longer(Enjoy:Adult, names_to = "variable") %>%
  group_by(variable) %>%
  summarise(prop = sum(value)/n(),
            n = n()) %>%
  arrange(desc(prop))
```

```
## # A tibble: 5 x 3
##   variable  prop     n
##   <chr>    <dbl> <int>
## 1 Enjoy    0.612  2892
## 2 Logical  0.493  2892
## 3 Adult    0.463  2892
## 4 Job      0.405  2892
## 5 Useful   0.402  2892
```

References

Hallquist, M. N., & Wiley, J. F. (2018). MplusAutomation: An R Package for Facilitating Large-Scale Latent Variable Analyses in Mplus. Structural equation modeling: a multidisciplinary journal, 25(4), 621-638.

Muthén, L.K. and Muthén, B.O. (1998-2017). Mplus User's Guide. Eighth Edition. Los Angeles, CA: Muthén & Muthén

R Core Team (2017). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/

Wickham et al., (2019). Welcome to the tidyverse. Journal of Open Source Software, 4(43), 1686, https://doi.org/10.21105/joss.01686

UC SANTA BARBARA