Introduction to R and RStudio

MM4DBER Training Team

Updated: August 29, 2023

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.

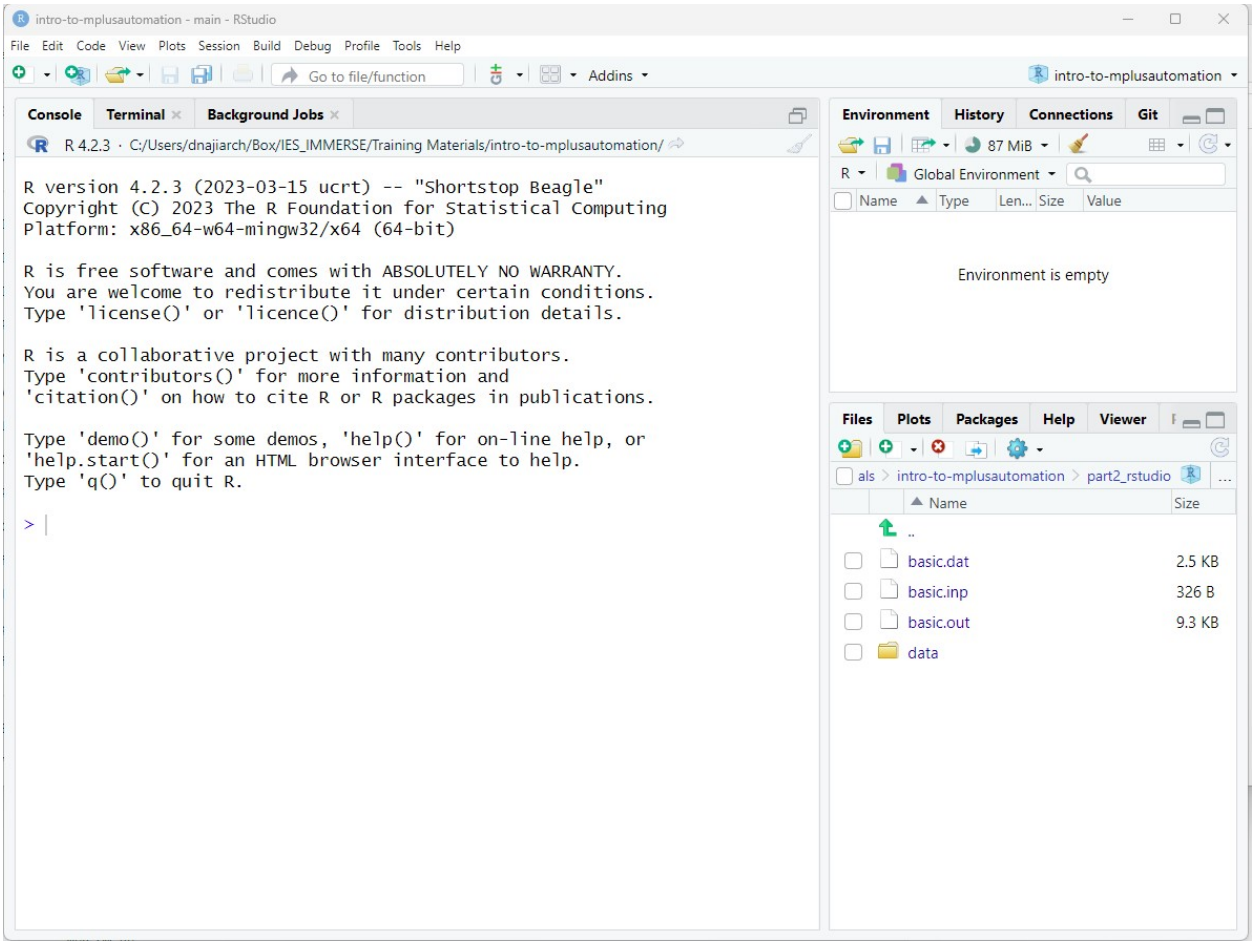
- Please visit our website to learn more and apply for the year-long fellowship.
- Follow us on Twitter!

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

Introduction:

- 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*.

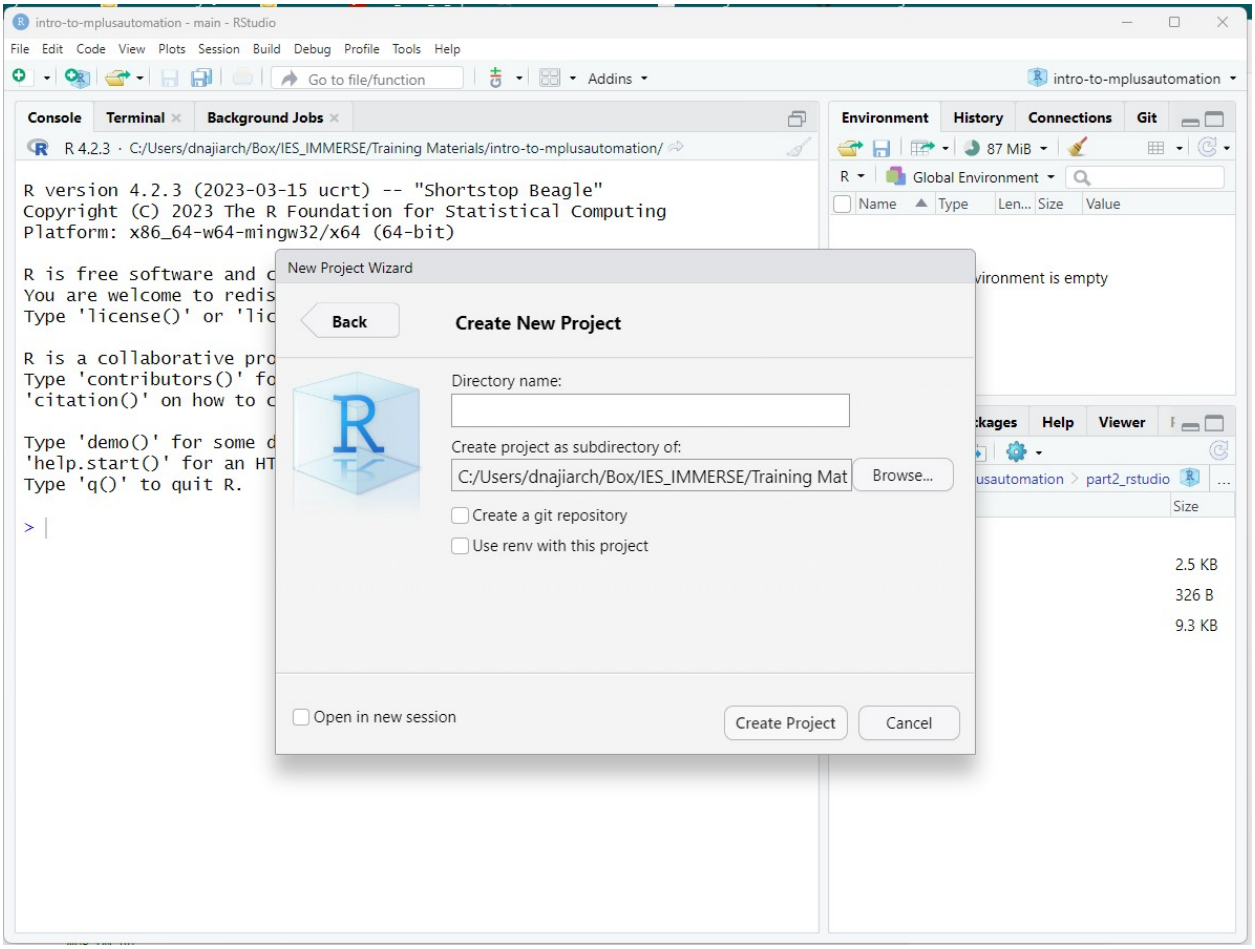
Step 1: Open RStudio



Open R studio on your desktop.

IMPORTANT: Because we are using a package that communicates with Mplus, we *must* use have Mplus installed to run Rstudio.

Step 2: Create a new R-project



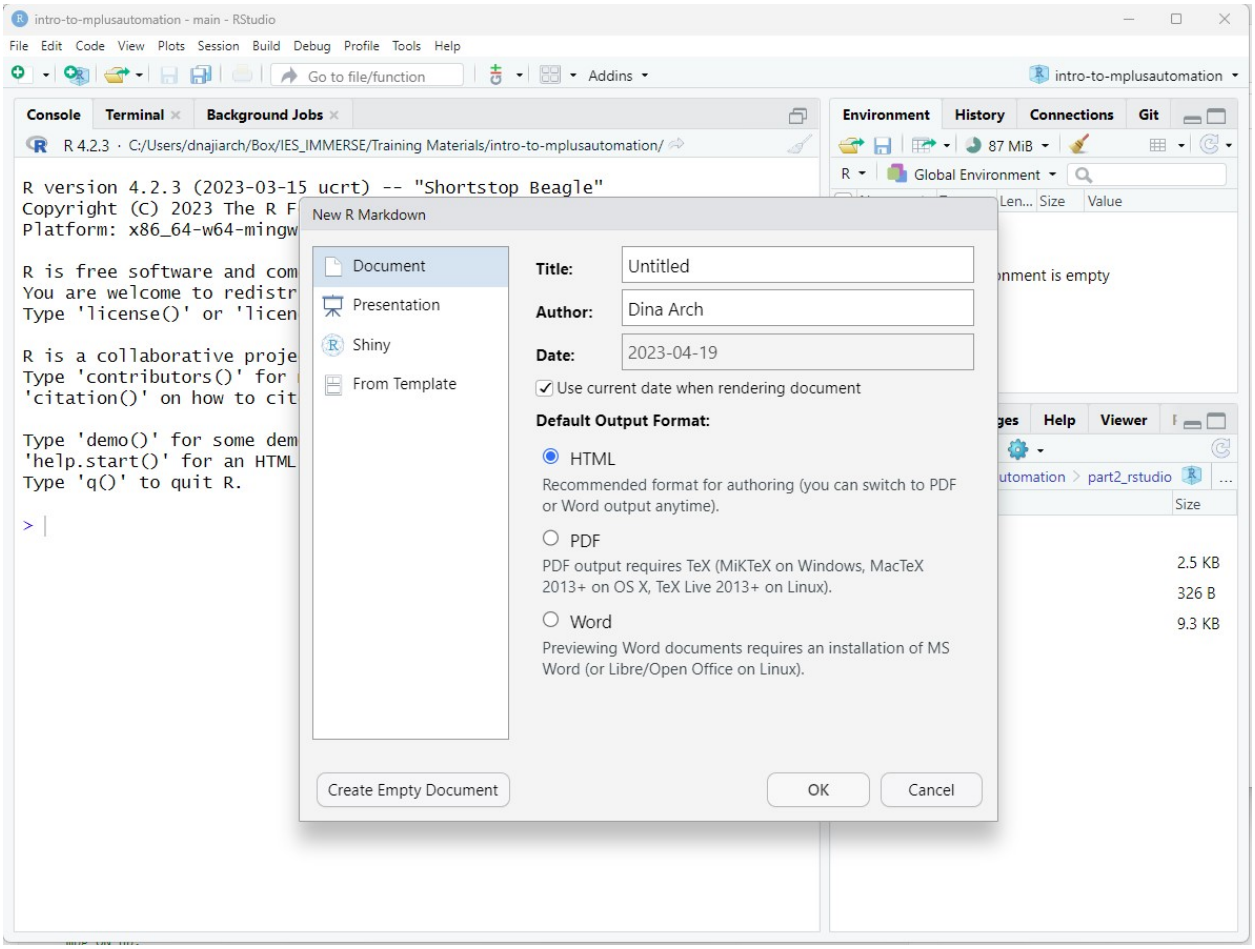
R-projects help us organize our folders , filepaths, and scripts. To create a new R project:

0. File –> New Project...
1. Click "New Directory" –> New Project –> Name your project (e.g., "introR-day5")

NOTE ABOUT PROJECT LOCATION (IMPORTANT): Choose a location on your computer that is not in too many enclosing folders. If your file path is too long (longer than 90 characters), Mplus cuts off the file.path and you will receive an **error message**.

2. Click "Create Project" to save your project (It will save as a folder on your computer).
3. Copy all the materials found on Github into this new project folder you created.

Step 3: Create an R-markdown document



- `R-markdown`: The `R-markdown` format provides a platform for us to neatly share our data science results. It allows us to organize our reports using texts, figures, and R code.
- This document you are reading was made using R-markdown!
- Next we will create an R-markdown file and write script to run a `type=basic` analysis using the R package, `MplusAutomation`.

To create an R-markdown:

0. File –> New File –> R Markdown...
1. Give the R-markdown a title such as "**Introduction to Rstudio**" Click "OK." You should see a new markdown with some example text and code chunks.
2. We want a clean document to start off with so delete everything from line 10 down.
3. Go ahead and save this document.

Step 4: Load packages

- The first code chunk (lines 8-10) is call the r setup code chunk. This will set the defaults for your document. For now we will leave this as is.
- The next code chunk in any given markdown should be the packages you will be using.
- To insert a code chunk, either use the keyboard shortcut `ctrl + alt + i` OR click the green button with the letter `C` on it (top panel).
- There are a few packages we want 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

#install.packages("palmerpenguins")
library(palmerpenguins) # data for plot example
```

Common error message types: [E.g.,. . .]

- If a function does not work and you receive an error message: `could not find function "xxx_function"`
- or if you try to load a package and you receive an error like this: `there is no package called` `` `xxx_package` ``
- then you will need to install the package using: `install.packages("xxx_package")`

  NOTE: 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(xxx_package)`, as shown in this document.

  The style of code we will use relies on the `tidyverse` package. 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.

Step 5: Make a quick plot (To show off what R can do!)

Look first

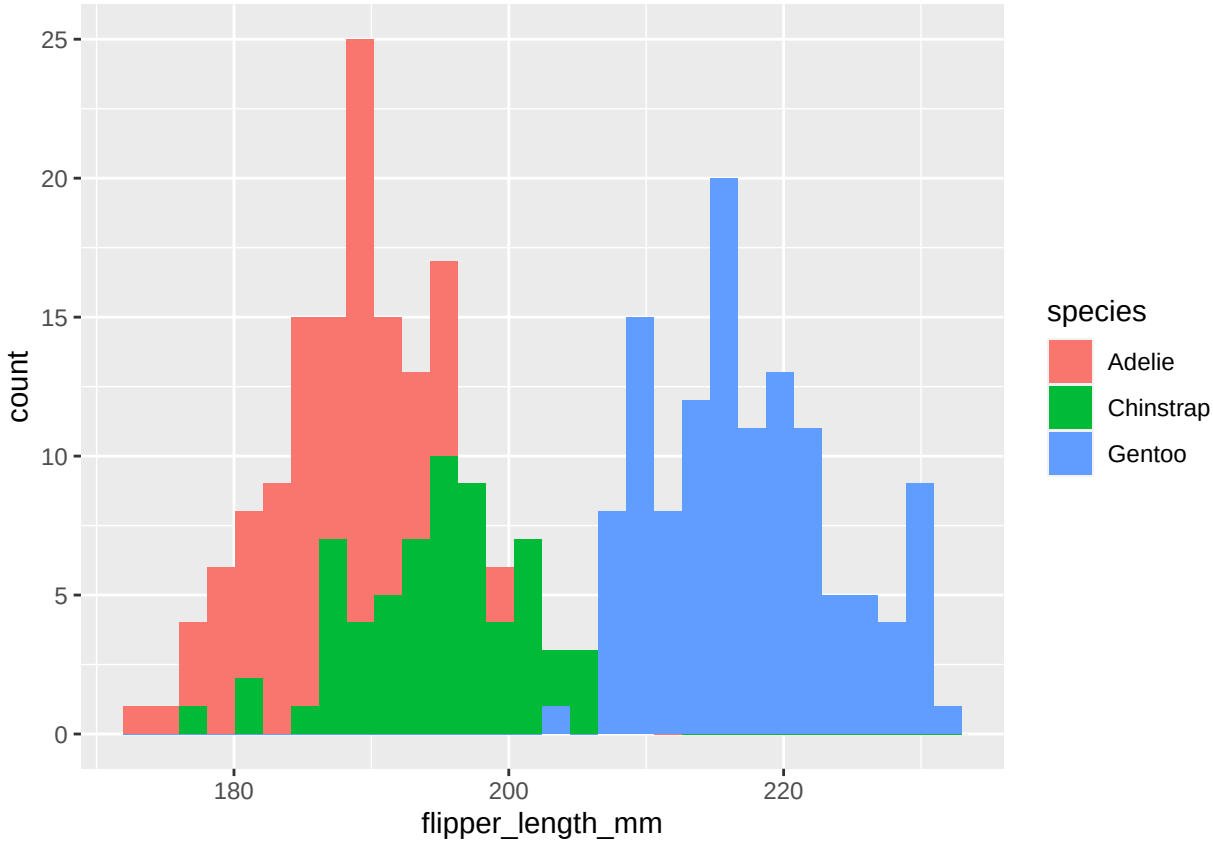
```
head(penguins)
```

```
## # A tibble: 6 x 8
##   species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
##   <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
## 1 Adelie  Torgersen           39.1          18.7               181        3750
## 2 Adelie  Torgersen           39.5          17.4               186        3800
## 3 Adelie  Torgersen           40.3          18                 195        3250
## 4 Adelie  Torgersen           NA            NA                  NA          NA
```

```
## 5 Adelie  Torgersen           36.7          19.3               193        3450
## 6 Adelie  Torgersen           39.3          20.6               190        3650
## # i 2 more variables: sex <fct>, year <int>
```

Plot!

```
ggplot(data = penguins, aes(x = flipper_length_mm, fill = species)) +
  geom_histogram(position = "identity")
```




```
# Extras to make plot pretty
# 1.  alpha = 0.5,
# 2.  scale_fill_manual(values = c("darkorange","purple","cyan4"))
# 3.  labs(x = "Flipper length (mm)", y = "Frequency")
# 4.  theme_minimal()
```

Explore data

Step 6: 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
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 5 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 7: 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 8: 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