Getting Started & Running a Simple Model

A Course in `MplusAutomation`

Adam Garber

October 22, 2021

Outline

1. Create a new `R Project`
2. Download a project repository from Github
3. Install & load packages
4. Read in data to R
5. View metadata (from SPSS files)
6. Prepare data
7. Write `.sav` , `.csv` , and `.dat` files
8. Introduce the "mplusObject Method"
9. Run a first model using `MplusAutomation`

Getting started repository:

https://github.com/garberadamc/QF-Getting-Started

Load packages

```
# install.packages("MplusAutomation")

library(MplusAutomation)
library(tidyverse)
library(haven)
library(here)
library(sjPlot)
```

Read in data

```
# object_name <- function_1("dataset_name.sav")

exp_data <- read_spss("https://garberadamc.github.io/project-site/data/explore_lab_data.sav")
```

View dataframe with labels & response scale meta-data

Note: Use the "print" option to save a PDF as a codebook containing metadata.

```
# the {haven} package stores the meta-data from SPSS files

# package_name::function_within_package()

sjPlot::view_df(exp_data)
```

Types of data for different tasks

- `.sav` (e.g., `spss_data.sav`): this data format is for SPSS files & contains variable labels (contains labels or meta-data)
- `.csv` (e.g., `r_ready_data.csv`): preferable data format for reading into `R` (non-labeled data)
- `.dat` (e.g., `mplus_data.dat`): this is the data format used to read into Mplus (no column names or strings)

Writing, reading, and converting data between 3 formats

Location, location, location!

NOTE: default directory in an `Rproject` is the "top-most" project folder

```
here()
```

```
## [1] "/Users/agarber/github/Quant-Fish-END"
```

Prepare data: Remove SPSS labels

Write a `.csv` data file (preferable format for reading into R)

```
# write_csv(data_name, here("sub_folder", data_name.csv"))

write_csv(exp_data, here("02-run-models", "data", "exp_data.csv"))
```

Read the unlabeled `.csv` data back into R

```
# new_data_name <- read_csv(here("sub_folder", "data_name.csv"))

nolabel_data <- read_csv(here("02-run-models", "data", "exp_data.csv"))
```

Write a `.dat` file using the `prepareMplusData()`

NOTE: This function removes the column header row and converts missing values to a period (.)

```
# prepareMplusData(new_data_name, here("sub_folder", "data_name.dat"))

prepareMplusData(nolabel_data, here("02-run-models", "data", "exp_data.dat"))
```

Function `prepareMplusData()`:

1. This function prints a minimal template of input syntax to start writing an Mplus input file.
2. By default missing values in your R object (`NA`) are converted to a period (`.`).

Preparing column-names to be `MplusAutomation` ready

Task: Make all variable names fit within the 8-character name limit (`Mplus`) while avoiding duplicates.

Rename columns manually

```
new_names <- nolabel_data %>%
  rename( motiv1 = item1 ,  # new_name = old_name
          motiv2 = item2 ,
          motiv3 = item3 ,
          comp1  = item4 ,
          comp2  = item5 ,
          comp3  = item6 ,
          belif1 = item7 ,
          belif2 = item8 ,
          belif3 = item9 )
```

Introduction to the "`mplusObject()` Method"

What does the `mplusObject()` function do?

- Takes an `R data.frame` and produces an object that contains all the information necessary to generate an Mplus **input** file.

What does the `mplusModeler()` function do?

1. It generates a data file (`.dat`)
2. It generates a input file (`.inp`)
3. It commands Mplus to **run** or estimates the model producing the output file (`.out`).

NOTE: Within the `mplusObject()` function there is a mix of `R` & `Mplus` syntax.

`R` terminology - `functions` & `arguments`

- `mplusObject()` is a function from the {`MplusAutomation`} package (i.e., `MplusAutomation::mplusObject()`)
- If preferred you can mention the package explicitly for greater transparency (i.e., `MplusAutomation::mplusObject()`)
- Functions have one or more `arguments` or `inputs`
- The inputs for the `mplusObject()` function include `TITLE =`, `VARIABLE =`, `ANALYSIS =`, `usevariables =`, `rdata =` (among others)
- Arguments within functions are separated by a comma (`,`)

Within an `mplusObject()`:

- **Black colored text** = Arguments or inputs (i.e., `R` code)
- **Green colored text** (within quotation marks) = `Mplus` syntax (e.g., `"type = basic;"`)

Create a template for `mplusObject()` & `mplusModeler()` functions

```
m_template  <- mplusObject(

  TITLE =
    "",

  VARIABLE =
    "",

  ANALYSIS =
    "",

  PLOT =
    "",

  OUTPUT =
    "",

  usevariables = colnames(),
  rdata =  )

m_template_fit <- mplusModeler(m_template,
                  dataout=here("", ".dat"),
                  modelout=here("", ".inp"),
                  check=TRUE, run = TRUE, hashfilename = FALSE)
```

Run a first model using the `mplusObject()` method

Model is `type = BASIC;` (i.e., returns descriptive statistics)

```
m_basic  <- mplusObject(

  TITLE = "PRACTICE 01 - Explore TYPE = BASIC",

  VARIABLE =
 "usevar=
  item1 item2 item3 item4 item5
  item6 item7 item8 item9 female;

  ! use exclamation symbol to make comments, reminders, or annotations in Mplus files ",

  ANALYSIS =
 "type = basic; ",

  usevariables = colnames(nolabel_data),
  rdata = nolabel_data)

m_basic_fit <- mplusModeler(m_basic,
               dataout=here("02-run-models", "mplus_files", "basic.dat"),
               modelout=here("02-run-models", "mplus_files", "basic.inp"),
               check=TRUE, run = TRUE, hashfilename = FALSE)
```

Always check your model!

- In the RStudio window pane on the `bottom-right`under the `files` tab click on the `mplus_files` folder
- There should be 3 new files in this location that were produced by `mplusModeler()`
- Click on the output file (`.out`) to check if the model estimated or if there are any error messages

---

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

Further resources & examples here:

https://garberadamc.github.io/project-site/

https://www.adam-garber.com/
