harmony package in R (V6)

## Versions

These instructions were written with Mplus Version 8.5 and R version 4.0.3.

## Get started with R

A nice tutorial for R beginners: <https://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf>

### Install and load the *harmony* package

To install and load the *harmony* package, enter the following commands in RStudio console (or in the R script) and execute them one at a time.

**install.packages("devtools")**

**library(devtools)**

**devtools::install\_github("hai-mn/harmony", dependencies = TRUE)**

**library(harmony)**

Note that once a package is installed, it does not need to be installed again. However, after a package is installed, in order to use its functionalities, the user needs to load it into memory by executing the library command each time a new R session is started.

The installation and loading of *devtools* reflects the fact that the *harmony* package requires some other dependent packages. You will notice that multiple R packages ( *tidyverse*, , *data.table*, *readxl*, *openxlsx*, *latex2exp*, *scales*, *ggpubr*, etc.) will be installed and loaded. The connection of these tools (packages) to *harmony* is fulfilled by setting the optional argument ‘dependencies’ to TRUE when installing *harmony* from github. During the installation of *harmony*, the user may be asked to update some of the packages to versions *harmony* requires, please do so and select 1. ALL to ensure successful installation.

### Set the working directory

The working directory is a folder in which the tables and plots generated by R are saved/stored. The default working directory in R can be found using the function getwd(). You can set a different working directory where your Mplus output file is located by using the following command. Note that regardless of your operating system, the path should be separated by /.

**setwd(“path/to/your/working/directory”)**

For instance, the command might look like: setwd("C:/Documents/my/working/directory") in Windows. setwd("/Users/Documents/my/working/directory") in Mac.

### Ensure that you have the required Mplus output and eliminate certain phrases in the comments to run *harmony*

Mplus has an ALIGN option to request extra output when running the alignment model (OUTPUT: ALIGN;). Ensure that you have included this option, and that your Mplus output contains the following sections:

1. INPUT INSTRUCTIONS
2. SUMMARY OF ANALYSIS
3. MODEL FIT INFORMATION
4. MODEL RESULTS
5. APPROXIMATE MEASUREMENT INVARIANCE (NONINVARIANCE) FOR GROUPS
6. ALIGNMENT OUTPUT
7. SAVEDATA INFORMATION

Because *harmony* searches for certain terms to parse the output, you should avoid or limit comments (following !) with certain keywords in the Mplus syntax. Words and phrases to avoid within comments include:

* class or classes
* knownclass
* continuous latent variables
* binary and ordered categorial
* UNIVARIATE PROPORTIONS AND COUNTS FOR CATEGORICAL VARIABLES
* threshold
* loadings
* Approximate Invariance Was Not Found For This Parameter

### Follow a step by step illustration of the tutorial example

#### Call functions in order

Note that when following this tutorial example (or when using *harmony* for the user’s own application) the user should follow the order of functions below. Doing so is needed because some commands expect output from earlier commands. After calling the functions in *harmony*, the names of the files generated and the folder where the files are saved will be printed in the console.

#### Generate basic tables listing the loadings, thresholds and alignment information

Basic tables listing loadings, thresholds, and alignment information can be generated with the following function:

**alignmentout(infile = " ", directory = " ")**

The user needs to enclose in quotation marks the path to the Mplus output file with the .out extension and folder name to store the output files in the working directory. For instance:

In Windows: alignmentout(infile ="c:/Documents/Mplus-output/Tutorial-Alignment-Fixed.out", directory = "Tutorial Output").

In Mac: alignmentout(infile ="/Users/Documents/Mplus-output/Tutorial-Alignment-Fixed.out", directory = "Tutorial Output").

If the Mplus output file resides in the working directory, you can abbreviate to: alignmentout(infile ="Tutorial-Alignment-Fixed.out", directory = “Tutorial Ouput”) in either operation system.

Immediately after the alignmentout function is run, basic information extracted from the output will be printed in the console, including the number and names of groups, number and names of factors, number and names of items, and number of categories and thresholds in each item.

The argument directory is optional. Thus, if the user doesn’t specify a folder name for it— e.g., alignmentout(infile ="Tutorial-Alignment-Fixed.out")—*harmony* creates a new folder named ‘Output\_current date’ in the working directory, such as Output\_2021-01-01 if *harmony* was run on January 1, 2021. This folder contains one csv file for loadings and multiple csv files for thresholds. An Excel file named “alignment\_tables.xlsx” is also created, although redundant this file is offered for convenience by placing the .csv files in a single file but on different tabs. Another folder named Misc will be created in the output folder, saving “cuts” (.txt files) of portions of the Mplus output after calling the alignmentout function.

The number of threshold csv files is one less than the number of categories of your items. In the tutorial example, because the items have three categories, there are two threshold csv files. If the items in your application had a different number of categories, a different number of threshold.csv files will be created (e.g., if your items had five categories, there would be four threshold csv files). In other words, in the tutorial example, the newly created folder contains three csv files: “threshold1.csv”, “threshold2.csv”, “loadings.csv”, along with an xlsx file, “alignment\_tables.xlsx”.

Also note that *harmony* can accommodate items with different numbers of categories. The number of csv files created will be one less than the largest number of categories. For instance, if a dataset contains a mixture of items, some with three and some with four categories, then three threshold.csv files will be created. Information from all items will be listed in the first two threshold csv files. The third csv file will have rows for each item, but the contents will be blank for items with three rather than four categories.

#### Generate alignment plots

In the *harmony* package, the alignmentthresholdplot and alignmentloadingplot functions will produce plots of thresholds and loadings by groups, as shown in Figures 2 and 3 of the main tutorial paper.

**alignmentthresholdplot(labelfile=" "**, directory = ""**)**

**alignmentloadingplot(labelfile=" "**, directory = ""**)**

The argument of the functions (labelfile) is optional, containing the path to a label file, supplying this file allows *harmony* to provide meaningful labels in a legend of the loadings and thresholds graphs. For instance:

In Windows: alignmentthresholdplot(labelfile="c:/Documents/group label.xlsx", directory = “Tutorial Output”).

In Mac: alignmentthresholdplot(labelfile="/Users/Documents/group label.xlsx", directory = “Tutorial Output”).

If the label file resides in the working directory: alignmentthresholdplot(labelfile="group label.xlsx", directory = “Tutorial Output”).

Note that *harmony* will use the thresholds and loadings csv files previously generated by the alignmentout function and located in the output folder that is specified by the user or automatically created by harmony (i.e., ‘Output\_current date’ ). The R syntax for the latter case is **alignmentthresholdplot(labelfile=" ").** Thus, it is important to run the functions in order and specify the output folder containing the thresholds and loadings of interest.

We first illustrate how to use such labels in our tutorial example and then explain how to create such a file for a user’s own application

The tutorial supplied label file “group label.xlsx” contains two columns, the first named GroupNumber (with values of 1, 2, 3, 4, 5, 6 down the rows) and the second named GroupLabel (with names Latinx\_Male, Black\_Male, White\_Male, Latinx\_Female, Black\_Female, and White\_Female down the rows). Additionally, 3 reshaped data files in a csv format (“threshold1\_longform.csv”, “thresholds2\_longform.csv”, and “loadings\_longform.csv”) will be added to the Misc folder. The user may customize and create their own visualizations of thresholds and loadings for alignment models with those datafiles. Note that, in the example, thresholds and loadings for lclass 7 denote the weighted averages.

For a user’s application, the user should create an Excel spreadsheet listing the group numbers and their labels. To do so, the user should first open a blank workbook in Excel. Then, name the first column GroupNumber(in its first row). In subsequent rows of the first column enter the numeric numbers used in the Mplus syntax. Name the second column GroupLabel. In subsequent rows, enter the desired labels corresponding to each group number from column one. Be sure to save the file in .xlsx format.

Similar to the alignmentout function, if the items in your application have a different number of categories, a different number of threshold\_longform.csv files will be created (i.e., the number of categories less one). Note that thresholds and loadings for the last value of lclass denote the weighted averages.

If your dataset contains items with mixed numbers of categories, some plots will have more items than other plots, and, comparison of thresholds across items that have different numbers of categories may not be meaningful.

#### Convert Item Factor Analysis (IFA) estimates to Item Response Theory (IRT) metric

To translates the parameter estimates from the IFA to IRT metric (i.e., converting loadings to discriminations and thresholds to difficulties) the user should call the convert2irt function:

**convert2irt()**

**convert2irt(directory = "")**

Note that this function can be called either without specific arguments/parameters (i.e., with empty parentheses following the function name) or with the argument directory where the name of the output folder is specified. The function creates datafiles that are saved in .csv format, including one csv file for the factor means and variances by group, one csv file for calculated discrimination, and multiple csv files for difficulties. Recall that the number of difficulties are equal to that of thresholds, so that the number of difficulties files are the number of categories less one. In the tutorial example, four csv file are created: “group\_factor means and variances.csv”, “discriminations.csv”, “difficulty1.csv”, and “difficulty2.csv”.

#### Generate cumulative and category probability curves

The function cpcCPC utilizes the previously calculated discriminations and difficulties and produces probability curves and cumulative category probability curves:

**cpcCPC (selected.item=" ", selected.group=" ", directory = "")**

The user needs to specify: (1), the item of interest, and (2), groups of interest separated by commas , to be plotted as shown in Figure4 of the manuscript. For instance,

cpcCPC(selected.item="BPI14", selected.group="3, 6", directory = "Tutorial Output")

Similarly, the folder name containing the IRT metrics should be specified as the directory. If the user chooses to store output files in the folder created by harmony, the argument directory should be omitted in the syntax (e.g., cpcCPC(selected.item="BPI14", selected.group="3, 6")).Generally speaking, we recommend that the user consistently specify the directory throughout the functions in harmony. The program allows a single item and one or more groups to be selected.

#### Generate value grid of the category and cumulative probability curve for selected item and group

cpc.csvexport function offers the users the flexibility to create their own probability curves upon their preferences.

**cpc.csvexport(selected.item=" ", selected.group=" ",** directory = ""**)**

One item and one group can be specified at a time. A datafile in csv format containing latent factor/trait values in IRT model (aka Theta), category probabilities and cumulative probabilities for the selected item and group will be created. If the user would like to obtain such datafiles for multiple items and groups, an example of R codes are provided.

In the csv datafile, the first column is a series values of Theta spans from -4 to 4 by 0.1 and the rest of the columns are the corresponding calculated probabilities. Numbers following cpc stands for the category probability and the number of such columns is equal to the number of categories in the item. Numbers following CPC stands for the cumulative probability and the number of such columns is equal to one less the number of categories in the item.

#### Detect small cell size in item categories by group

**cellsizedetect(**infile=" ", n.detect="0", silent = TRUE**)**

cellsizedetect function is developed to detect potential small numbers of item responses in groups. Multiple crosstabs between the group and item categories will be generated behind the scenes and small cell sizes will be identified. An Mplus output containing a “CROSSTABS FOR CATEGORICAL VARIABLES” section is required to call this function. The user may find a sample Mplus input file named “basic-crosstabs.inp“.

An Mplus output file (e.g., basic-crosstabs.out) should be provided for the first argument infile, number of cell size intended to be detected needs to be specified for the argument n.detect (e.g., if a number of 10 is entered, categories in items by group which are equal to or smaller than 10 will be printed in the console), and silent = TRUE suppresses printouts of all crosstabs generated behind the scenes.

#### An example

cellsizedetect(infile="basic-crosstabs.out", n.detect="0", silent = TRUE)

Here we provided a Mplus output file "basic-crosstabs.out" in the infile and the zero number of cell size in n.detect argument, i.e. we want to detect in the crosstabs. silent = TRUE (default option) which means crosstabs will not print out.

In order to run cellsizedetect function, Mplus output should have a "CROSSTABS FOR CATEGORICAL VARIABLES" section.

#### Other R packages for the alignment model

At the time harmony was developed, we were aware of one other R package that extracted results of alignment models from Mplus output files:

 (MIE)(<https://github.com/MaksimRudnev/MIE.package>)

This MIE package was designed for continuous items, however, and couldn’t accommodate analyses with categorical items. harmony filled the latter need.