MplusAutomation:

An *R* package for Enhancing Latent Variable Modeling using *Mplus*

Michael Hallquist
University of Pittsburgh

and

Joshua Wiley
University of California – Los Angeles

Introduction

- MplusAutomation is a package for *R*, a leading open-source statistics programming language.
- The package overcomes some of the pragmatic limitations of Mplus to streamline the process of building and testing latent variable models.
  - The package also tries to facilitate large-scale, complex projects (e.g., Monte Carlo simulation studies) that require organized access to numerous model outputs.
- MplusAutomation seeks to extend, not duplicate, the functionality of Mplus.
- More comprehensive documentation provided at:

  http://cran.r-project.org/web/packages/MplusAutomation/vignettes/Vignette.pdf

Clarifications and Caveats

- MplusAutomation is first and foremost a text-processing engine.
  - Functions parse and import Mplus outputs into *R* data structures.
- MplusAutomation does not have access to the internals of Mplus.
- Each section of output to be extracted typically requires custom handling.
  - Thus, functionality of the package has grown slowly as users have requested features.

Overview

- Rationale and introduction to MplusAutomation
- Brief Example: Latent class analysis versus confirmatory factor analysis.
- Dynamic models and documents with *R*.
- Simulation example: Effect of scatter on clustering.
  - Generating non-standard datasets using R.
  - Managing simulation datasets and syntax files
  - External Monte Carlo study with Mplus.

Strengths of Mplus

- Highly flexible latent variable modeling
- Simple, powerful syntax
- Intelligent parameter defaults
- Accessible Bayesian estimation
- “One ring to rule them all”
  - Mplus can estimate nearly any regression or latent variable model.
- Excellent support for applied researchers.

Weaknesses of Mplus

- Separate syntax files for each model.
- Not easy to run large groups of models.
- Difficult to compare model outputs.
  - Lengthy text files often cannot be indexed quickly by eye.
- Results are often embedded both in the output file and unlabeled external files.

Objectives of MplusAutomation

- Integrate Mplus, the premier latent variable modeling program, with *R*, a robust statistics language.
  - *R* has many packages for data analysis and visualization.

1. Provide tools to create and manage related models.
   - Modeling projects (esp. simulation) often involve the creation of tens or hundreds of related model specifications that differ in subtle ways.
2. Automate estimation of model batches.
3. Provide tools to extract and compare model fit statistics and parameter estimates.
   - Import Mplus output into *R* conventions, e.g., *data.frame*.
4. Synthesis: provide a pipeline for fully integrating

1. Create and manage related models

- Examples:
  - Same model, different outcome (e.g., genetics studies)
  - Same model, different data (e.g., Monte Carlo)
  - Same data, different parameterization (e.g., Poisson, ZIP, NB growth)
- MplusAutomation provides a rudimentary macro language that specifies how Mplus syntax should vary across model input files.
- createModels(templatefile)
  - templatefile: the name of an Mplus template file to be processed into multiple input files.
- Re-running createModels() overwrites all input files related to the template file. Allows for tweaking of model.
- createModels can be used to create well-named, well-structured directories for relevant Mplus syntax.

Mplus template basics

- All tags use double-brackets: [[tag]]
- Templates consist of init and body sections.
- Init definitions:
  1. iterators: one or more looping variables that define how the input files will be created.
  2. filename: defines the filenames for input files to be created. In order to ensure unique names, tags are typically included in the filename definition. Example: “[[classes]]-class model.inp”
  3. outputDirectory: the directory where input files will be saved. Preferable to use absolute directory. Can specify a subdirectory structure using tags. Example: “C:/Data/[[outcome]]/[[model]]”

Mplus template basics

- Tag types:
  1. Simple: the value defined in init is substituted wherever the tag occurs.
  2. List: linked to an iterator, the appropriate list element is substituted depending on the iterator value.
  3. Iterator: the numeric value of the iterator is substituted.
  4. Conditional: A block of Mplus syntax is included only if a condition is met.

Mplus template file example

```
[[init]]

iterators = model;
model=1:5;
modelnames#model = CFA LCA2 LCA3 LCA4 LCA5;
title#model = "Confirmatory Factor Analysis"
                        "2-class LCA"
                        "3-class LCA"
                        "4-class LCA"
                        "5-class LCA";
classes#model = 0 2 3 4 5;

filename = "[[modelnames#model]].inp";
outputDirectory = templateExample/[[modelnames#model]];

[[/init]]
```

Mplus template file example

```
TITLE: [[title#model]]
DATA:   FILE = "../ex7.4.dat";
VARIABLE: NAMES = u1 u2 u3 u4; MISSING=.;
          CATEGORICAL = u1 u2 u3 u4;

[[model > 1]]  CLASSES = c([[classes#model]]);  [[/model > 1]]

ANALYSIS:
[[model == 1]]  TYPE=GENERAL; ESTIMATOR=MLR;  [[/model == 1]]
[[model > 1]]   TYPE = MIXTURE;  STARTS = 250 25;  [[/model > 1]]

MODEL:
[[model==1]]   factor BY u1-u4; [[/model==1]]
[[model > 1]]  %OVERALL%  [[/model > 1]]

OUTPUT: TECH1 TECH8;
```

Create model input files

```
> library(MplusAutomation)
Loading required package: tcltk
Loading required package: plyr
Loading required package: boot

> createModels("templateExample.txt")
[1] "writing file:  CFA.inp"
[1] "writing file:  LCA2.inp"
[1] "writing file:  LCA3.inp"
[1] "writing file:  LCA4.inp"
[1] "writing file:  LCA5.inp"
```

2. Estimation automation

- runModels(directory, recursive, showOutput, replaceOutfile, logFile, Mplus_command)
  - directory: directory path containing all input files to be run. Default: getwd()
  - recursive: (TRUE/FALSE) run input files in subdirs.
  - showOutput: (TRUE/FALSE) show TECH8 progress in *R*.
  - replaceOutfile: whether to re-run a model that has existing output. Options are “always”, “never”, or “modifiedDate”. Default: “always”.
  - logFile: Name of file to store batch file progress. Default: “Mplus Run Models.log”
  - Mplus_command: command to be used to run Mplus.
- See also runModels_Interactive()

Run our template file

```
> library(MplusAutomation)
> getwd()
[1] "/Users/michael/MplusAutomation/products/M3_May2013”

> runModels(directory="templateExample", recursive=TRUE)

Running model: CFA.inp
System command: cd "templateExample/CFA" && "/Applications/Mplus/mplus" "CFA.inp"

Running model: LCA2.inp
System command: cd "templateExample/LCA2" && "/Applications/Mplus/mplus" "LCA2.inp"

Running model: LCA3.inp
System command: cd "templateExample/LCA3" && "/Applications/Mplus/mplus" "LCA3.inp"

Running model: LCA4.inp
System command: cd "templateExample/LCA4" && "/Applications/Mplus/mplus" "LCA4.inp"

Running model: LCA5.inp
System command: cd "templateExample/LCA5" && "/Applications/Mplus/mplus" "LCA5.inp"
```

3. Extracting output

- allModels = readModels(target, recursive, filefilter)
  - target: the output file or directory from which to read.
  - recursive: (TRUE/FALSE) whether to search subdirs.
  - filefilter: a Perl-compatible regular expression that filters output files of interest.
- Returns an *R* list where each element corresponds to a single Mplus output file.
- For each output file, readModels attempts to extract all available information, including summary statistics, model parameter estimates, and confidence intervals.
- For MONTE CARLO and IMPUTATION runs, also reads datasets for each replication.

3. Extracting output

- Where available, readModels will extract the following (and more!) from each output file:

```
$summaries:    model fit statistics, details
$parameters:   model results and CIs
$mod_indices:  model modification indices
$savedata:     savedata output from FILE=
$bparameters:  MCMC chains from BPARAMETERS=
$residuals:    model-estimated residuals
$class_counts: class counts/uncertainty
$gh5:          data generated from PLOT:
```

- Typical *R* techniques for tabulating output:

```
tscore_summaries <- do.call("plyr::rbind.fill",
      sapply(ClusANPD, "[", "summaries"))
```

Extract example outputs

```
> allModels <- readModels("templateExample", recursive=TRUE)
Reading model:  templateExample/CFA/CFA.out
Reading model:  templateExample/LCA2/LCA2.out
Reading model:  templateExample/LCA3/LCA3.out
Reading model:  templateExample/LCA4/LCA4.out
Reading model:  templateExample/LCA5/LCA5.out

> str(allModels, max.level=1)
List of 5
 $ templateExample.CFA.CFA.out  :List of 15
  ..- attr(*, "class")= chr [1:2] "list" "mplus.model"
  ..- attr(*, "filename")= chr "templateExample/CFA/CFA.out"
 $ templateExample.LCA2.LCA2.out:List of 15
  ..- attr(*, "class")= chr [1:2] "list" "mplus.model"
  ..- attr(*, "filename")= chr "templateExample/LCA2/LCA2.out"
 $ templateExample.LCA3.LCA3.out:List of 15
  ..- attr(*, "class")= chr [1:2] "list" "mplus.model"
  ..- attr(*, "filename")= chr "templateExample/LCA3/LCA3.out"
 $ templateExample.LCA4.LCA4.out:List of 15
  ..- attr(*, "class")= chr [1:2] "list" "mplus.model"
  ..- attr(*, "filename")= chr "templateExample/LCA4/LCA4.out"
 $ templateExample.LCA5.LCA5.out:List of 15
  ..- attr(*, "class")= chr [1:2] "list" "mplus.model"
  ..- attr(*, "filename")= chr "templateExample/LCA5/LCA5.out"
 - attr(*, "class")= chr [1:2] "list" "mplus.model.list"
```

Extract example outputs (cont.)

```
> str(allModels[["templateExample.CFA.CFA.out"]])
List of 15
 $ input          :List of 6
  ..$ title   : chr "Confirmatory Factor Analysis;"
  ..$ data    :List of 1
  .. ..$ file: chr "\"../ex7.4.dat\""
  ..$ variable:List of 3
  .. ..$ names      : chr "u1 u2 u3 u4"
  .. ..$ missing    : chr "."
  .. ..$ categorical: chr "u1 u2 u3 u4"
  ..$ analysis:List of 1
  .. ..$ type: chr "GENERAL"
  ..$ model   : chr [1:3] "" "     factor BY u1-u4;" ""
  ..$ output  : chr [1:4] " TECH1 TECH8;" "" "" ""
  ..- attr(*, "class")= chr [1:2] "list" "mplus.inp"
  ..- attr(*, "start.line")= int 6
  ..- attr(*, "end.line")= int 23
 $ warnings       :List of 1
  ..$ : chr [1:4] "TECH8 option is available only with analysis types MIXTURE, RANDOM,
or" "TWOLEVEL with estimators ML, MLF, or MLR or ALGORITHM=INTEGRATION." "Request for
TECH8 is ignored." "1 WARNING(S) FOUND IN THE INPUT INSTRUCTIONS"
  ..- attr(*, "class")= chr [1:2] "list" "mplus.warnings"
 $ errors         : list()
  ..- attr(*, "class")= chr [1:2] "list" "mplus.errors”
```

Extract example outputs (cont.)

```
 $ summaries      :Classes 'mplus.summaries' and 'data.frame':
   1 obs. of  21 variables:
  ..$ Mplus.version       : chr "7"
  ..$ Title               : chr "Confirmatory Factor Analysis;"
  ..$ AnalysisType        : chr "GENERAL"
  ..$ DataType            : chr "INDIVIDUAL"
  ..$ Estimator           : chr "WLSMV"
  ..$ Observations        : num 500
  ..$ Parameters          : num 8
  ..$ ChiSqM_Value        : num 2.19
  ..$ ChiSqM_DF           : num 2
  ..$ ChiSqM_PValue       : num 0.334
  ..$ ChiSqBaseline_Value : num 150
  ..$ ChiSqBaseline_DF    : num 6
  ..$ ChiSqBaseline_PValue: num 0
  ..$ CFI                 : num 0.999
  ..$ TLI                 : num 0.996
  ..$ RMSEA_Estimate      : num 0.014
  ..$ RMSEA_90CI_LB       : num 0
  ..$ RMSEA_90CI_UB       : num 0.091
  ..$ RMSEA_pLT05         : num 0.677
  ..$ WRMR                : num 0.36
  ..$ Filename            : chr "CFA.out"
```

Extract example outputs (cont.)

```
 $ parameters     :List of 2
  ..$ unstandardized: Classes 'mplus.params' and 'data.frame': 9 obs. of  6 variables:
  .. ..$ paramHeader: chr [1:9] "FACTOR.BY" "FACTOR.BY" "FACTOR.BY" "FACTOR.BY" ...
  .. ..$ param      : chr [1:9] "U1" "U2" "U3" "U4" ...
  .. ..$ est        : num [1:9] 1 0.914 -0.786 -0.672 0 0.005 0.01 0.136 0.46
  .. ..$ se         : num [1:9] 0 0.177 0.158 0.145 0.056 0.056 0.056 0.056 0.108
  .. ..$ est_se     : num [1:9] 999 5.17 -4.98 -4.64 0 ...
  .. ..$ pval       : num [1:9] 999 0 0 0 1 0.929 0.858 0.016 0
  .. ..- attr(*, "filename")= chr "templateExample/CFA/CFA.out"
  ..$ irt.parameterization: Classes 'mplus.params' and 'data.frame': 9 obs. of 6
variables:
  .. ..$ paramHeader: chr [1:9] "FACTOR.BY" "FACTOR.BY" "FACTOR.BY" "FACTOR.BY" ...
  .. ..$ param      : chr [1:9] "U1" "U2" "U3" "U4" ...
  .. ..$ est        : num [1:9] 0.923 0.79 -0.63 -0.512 0 0.008 -0.019 -0.298 1
  .. ..$ se         : num [1:9] 0.201 0.159 0.124 0.108 0.083 0.09 0.105 0.133 0
  .. ..$ est_se     : num [1:9] 4.59 4.96 -5.08 -4.76 0 ...
  .. ..$ pval       : num [1:9] 0 0 0 0 1 0.929 0.858 0.025 1
  .. ..- attr(*, "filename")= chr "templateExample/CFA/CFA.out"
  .. ..- attr(*, "probit")= chr "discrimination*(theta - difficulty)"
```

Summarizing results

- MplusAutomation also provides three related functions for visually comparing model summaries:
  - **`showSummaryTable`**: tabulate model summaries on screen. (requires Tcl/Tk)
  - **`HTMLSummaryTable`**: create HTML file with model summary table.
  - **`LatexSummaryTable`**: create model summary table as LaTex. (Useful for Sweave/knitr documents)

Summarizing results

```
> LatexSummaryTable(allModels, sortBy="BIC")
```

	Title	LL	Parameters	AIC	AICC	BIC
1	Confirmatory Factor Analysis	-1327.32	8.00	2670.63	2670.93	2704.35
2	2-class LCA	-1325.21	9.00	2668.43	2668.79	2706.36
3	3-class LCA	-1320.35	14.00	2668.70	2669.56	2727.70
4	4-class LCA	-1318.92	19.00	2675.84	2677.43	2755.92
5	5-class LCA	-1318.84	24.00	2685.68	2688.21	2786.83

Summarizing results

- Focal comparison: often useful to compare side-by-side model fit and parameter estimates between two models.
  - Introducing one parameter into a large SEM can affect many estimates, and it is often difficult to assess this visually.
- compareModels(m1, m2, show, equalityMargin, sort, showFixed, showNS, diffTest)
  - m1: the first model, generated by readModels()
  - m2: the second model
  - show: what aspects of the models should be compared.
    - c("all", "summaries", "equal", "diff", "pdiff", and "unique")
  - equalityMargin: Defines the discrepancy between models that is considered equal.
    - Different margins can be specified for p-value equality versus parameter equality. c(param=.05, pvalue=.01)
  - sort: how to sort the output of parameter comparisons.
    - c("none", "type", "alphabetical", and "maxDiff")
  - showFixed: whether to output constrained parameters: TRUE/FALSE
  - showNS: whether to output non-significant effects: TRUE/FALSE
  - diffTest: chi-square difference test for nested models: TRUE/FALSE

Summarizing results

- Constrain two factor loadings to equality
- Compare constrained versus unconstrained models.

```
MODEL:
   factor BY u1
          u2
          u3 (1)
          u4 (1);

> compareModels(
    allModels[["CFA_constrained.out"]],
    allModels[["CFA.out"]],
    show=c("summaries", "equal", "diff"),
    equalityMargin=.01, diffTest=TRUE)
```

Summarizing results

```
Mplus model comparison
----------------------
Model 1:  templateExample/CFA/CFA_constrained.out
Model 2:  templateExample/CFA/CFA.out
------

Model Summary Comparison
------------------------
             m1                                  m2
Title        CFA: Constrain u3 and u4 loadings to Confirmatory Factor Analysis
               equality
Observations 500                                 500
Estimator    MLR                                 MLR
Parameters   7                                   8
LL           -1327.578                           -1327.316
AIC          2669.156                            2670.633
BIC          2698.658                            2704.35

  MLR Chi-Square Difference Test for Nested Models Based on Loglikelihood
  -----------------------------------------------------------------------
  Difference Test Scaling Correction:  1.2585
  Chi-square difference:  0.4164
  Diff degrees of freedom:  1
  P-value:  0.5188
```

Summarizing results

```
Model parameter comparison
--------------------------
  Parameters present in both models
=========

  Equal in both models (param. est. diff <= 0.01)
  -----------------------------------------------
 paramHeader param m1_est m2_est . m1_se m2_se . m1_est_se m2_est_se . m1_pval m2_pval
  Thresholds  U1$1  0.000   0.00 | 0.127 0.128 |    -0.001    -0.003 |   0.999   0.998
  Thresholds  U2$1  0.011   0.01 | 0.119 0.119 |     0.088     0.087 |   0.930   0.931
  Thresholds  U3$1  0.019   0.02 | 0.106 0.110 |     0.180     0.180 |   0.857   0.857
  Thresholds  U4$1  0.258   0.25 | 0.107 0.105 |     2.404     2.391 |   0.016   0.017


  Parameter estimates that differ between models (param. est. diff > 0.01)
  ------------------------------------------------
 paramHeader  param m1_est m2_est . m1_se m2_se . m1_est_se m2_est_se . m1_pval m2_pval
   FACTOR.BY     U2  0.866  0.846 | 0.335 0.333 |     2.582     2.543 |   0.010   0.011
   FACTOR.BY     U3 -0.617 -0.680 | 0.176 0.216 |    -3.494    -3.151 |   0.000   0.002
   FACTOR.BY     U4 -0.617 -0.547 | 0.176 0.198 |    -3.494    -2.767 |   0.000   0.006
   Variances FACTOR  2.311  2.366 | 1.092 1.140 |     2.117     2.076 |   0.034   0.038
```

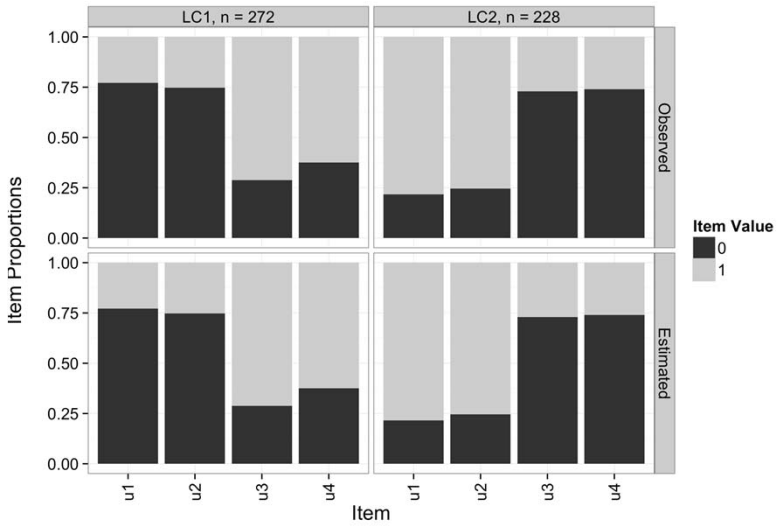
Bonus bonus: plotting in R

(in just 29 lines)

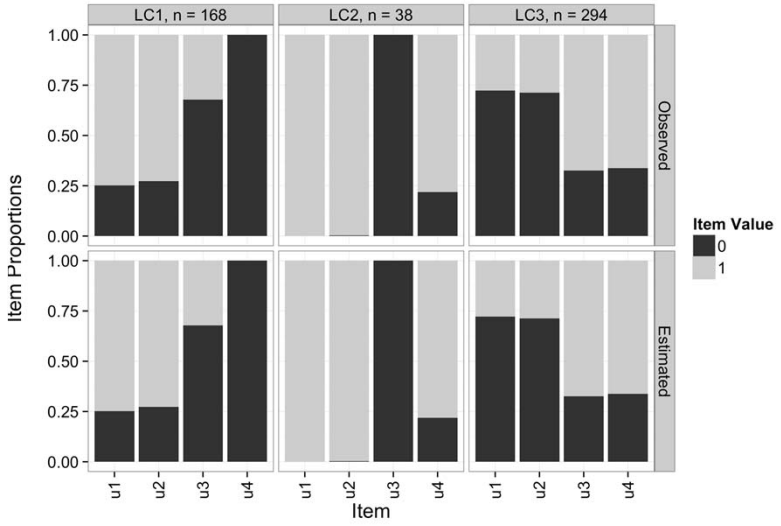
```
setwd(”~/MplusAutomation/products/M3_May2013/templateExample")
library(MplusAutomation)
lcaModels <- readModels(getwd(),recursive=TRUE,filefilter="LCA.*")

for (m in lcaModels) {
  numLC <- max(m$class_counts$modelEstimated$class)
  png(paste0("LCA", numLC, "Probs.png"), width=9, height=6,
units="in", res=300)
  plotLCAProbs(m, catValues=c(0,1))
  dev.off()
}
```

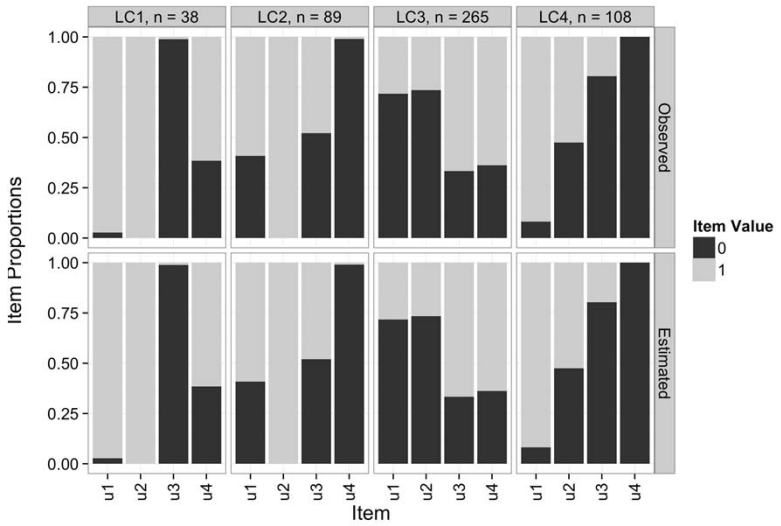
2-class LCA probabilities



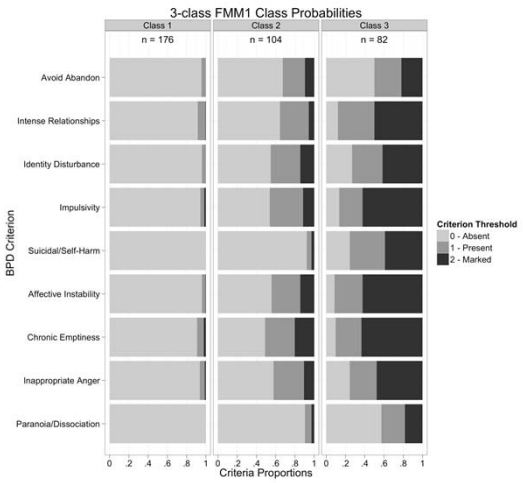
3-class LCA probabilities

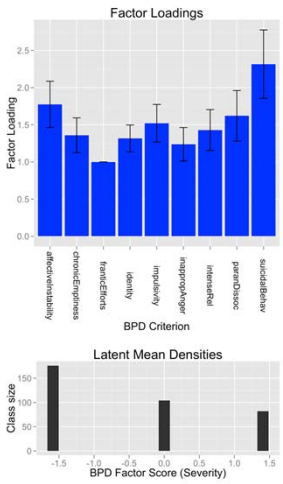


4-class LCA probabilities



Fancier plots possible





```
plotLCAProbs <- function(mplus_model, catValues) {
  require(ggplot2); require(reshape2)
  if (missing(catValues)) { stop("Must provide values of categorical items low-to-high") }

  estprobs <- mplus_model$gh5$means_and_variances_data$estimated_probs$values
  obsprobs <- mplus_model$gh5$means_and_variances_data$observed_probs$values

  estprobs <- melt(estprobs, varnames = c('param', 'LC'), value.name="prob", na.rm =
TRUE); estprobs$est <- 1
  obsprobs <- melt(obsprobs, varnames = c('param', 'LC'), value.name="prob", na.rm =
TRUE); obsprobs$est <- 0

  gh5probs <- rbind(estprobs, obsprobs)

  numCat <- length(catValues)
  numItems <- max(gh5probs$param)/numCat
  numLC <- max(gh5probs$LC)
```

```
  gh5probs$LC <- factor(paste0("LC", gh5probs$LC, ", n = ",
rep(mplus_model$class_counts$mostLikely$count, each=max(gh5probs$param))))
  gh5probs$uval <- factor(catValues)
  gh5probs$item <- factor(paste0("u", rep(1:numItems, each=numCat)))
  gh5probs$est <- factor(gh5probs$est, levels=c(0,1), labels=c("Observed", "Estimated"))

  g <- ggplot(gh5probs, aes(x=item, y=prob, fill=uval)) +
   geom_bar(stat="identity") + facet_grid(est~LC) +
   theme_bw(base_size=16) + ylab("Item Proportions\n") + xlab("Item")+
   theme(axis.text.x=element_text(angle=90, hjust=1, vjust=0.5)) +
   scale_fill_grey("Item Value", start=0.2, end=0.8)

  print(g)
}
```