Latent Regression in Higher-Order Item Response Theory with the R Package hlt

Michael J. Kleinsasser

Department of Biostatistics, University of Michigan, Ann Arbor, Michigan, USA

734-635-5608

mkleinsa@umich.edu

Ritesh Mistry

Department of Health Behavior and Health Education, University of Michigan, Ann Arbor, Michigan, USA

riteshm@umich.edu

Hsing-Fang Hsieh

Department of Health Behavior and Health Education, University of Michigan, Ann Arbor, Michigan, USA

Trivellore Raghunathan

Department of Biostatistics, University of Michigan, Ann Arbor, Michigan, USA

The National Cancer Institute of the National Institutes of Health under award number R01CA201415 supported the research reported in this paper. The Software is open source and publicly available under the GPL-2 license.

Item response theory (IRT) has become a standard method in the analysis of survey assessment data. In the IRT paradigm, the performance of test takers (i.e., their capacity to answer a question correctly) is explained by individual ability and the properties or characteristics of the test. Classically, ability is assumed to be 1) univariate and 2) homogenous across person characteristics. First, ability, which is a placement of each test taker on a continuum, can often be explained by heterogeneity among study participants. Second, recent applications involve characterizing a general, higher-order, ability which explain multiple, first-order, domains of ability. The natural progression, then, is to explain the heterogeneity of general ability. The open-source R package **hlt** implements the random-walk Metropolis-Hastings algorithm to estimate the higher-order IRT model by implementing a flexible Bayesian framework. We implement a higher-order generalize partial credit model and its extension of latent regression with the goal of explaining the relationship between the general latent construct and a set of explanatory variables. We provide the details about the model, estimation, software package, examples using simulated data, and an example data analysis from a real survey.

Item Response Theory, Bayesian data analysis, higher-order model, latent regression, hierarchical modeling, R package

Introduction

There are many applications in psychometrics and educational testing where a general latent domain, also called a factor or dimension, is sought to explain multiple underlying behaviors. Examples include the domains of the five-factor model of personality (Costa et al., 1995); quality of life among cancer patients (Gotay et al., 2002), and assessment of stress in cancer patients with the Perceived Stress Scale (Golden-Kreutz et al., 2004); and, in academic self-concept theory (Marsh & Hocevar, 1985), among many others.

A recent study examined demographic and social-cultural differences in family functioning within two urban Indian contexts (Hsieh et al., 2022). The study took a multi-dimensional approach to describe salient dimensions of family functioning. The results support a general latent factor of family functioning manifested by three primary dimensions. The authors conducted a latent regression analysis on the general latent factor and found that family functioning was predicted by gender, socio-economic status, and other social-cultural characteristics. In the analysis, the authors employed a higher-order item response theory model (de la Torre & Song, 2009), which employed the generalized partial credit model and a new prior structure which implements an extension of the prior work of de la Torre & Song (2009) to include person covariates via latent regression. The purpose of the present article is to demonstrate the statistical framework used in Hsieh et al. (2022), improvements and refinements to the model, and an open-source software tool for the statistical modeling.

Item response theory, or IRT (Cai et al., 2016), constitutes a class of statistical models which relate the ability of survey respondents to answer questions correctly and the varying difficulty, discrimination, and other item level criteria of a survey. The classical assumption of IRT models is that of univariate ability. In

this scenario, a single underlying trait drives the person level ability to perform on a test. One recent advancement in IRT involves changing the univariate ability assumption to more complicated factor structures where multiple related latent traits are formulated to explain underlying behavior. One such formulation provides a hierarchical structure to the factors where one second-order, or higher-order, factor is extracted from a set of domain level factors. This higher-order factor provides a general measure of ability which can characterize overall test performance and performance on domains in a single model.

Huang et al. (2013) provides an overview of five types of latent trait structures using in IRT beyond the unidimensional case. The authors describe: 1) the consecutive unidimensional (CU-IRT) approach where multiple traits are assessed by breaking the survey up and fitting the unidimensional model to each subset of the questions; 2) multidimensional structure (M-IRT) which introduces correlations between the latent traits of the consecutive approach; 3) the composite unidimensional (C-IRT) structure, which we call the unidimensional model, since it assumes a single factor over the multiple dimensions; 4) the bi-factor (B-IRT) where a single factor competes with multiple unidimensional factors, combining 1) and 3); and finally, 5) the higher-order (HO-IRT) factor structure, where a general ability informs the multiple dimensions measured by the survey. Although this article focuses on the HO-IRT model, it has been shown that HO-IRT parameters can be estimated as a constrained re-parameterization of the B-IRT model (Rijmen, 2010). Thus, there can be considerable overlap between these approaches.

Another useful extension to the univariate IRT framework involves regressing the latent ability on a set of explanatory factors (Wilson & De Boeck, 2004), which we refer to as latent regression IRT. This extension attempts to explain

differences, or heterogeneity, in ability by person level attributes such as gender or socioeconomic status. There are other IRT extensions which we did not explore, such as differential item functioning (DIF) and multiple group analysis, but the model and software discussed here form a framework which can be extended to incorporate these other extensions of IRT.

In this article, we refer to the higher-order IRT model accounting for measurement, but not latent regression, as descriptive HO-IRT, and the model with a latent regression component is referred to as explanatory HO-IRT. For succinctness, we also refer to higher-order item response theory model with latent regression as HO-IRT-LR, where LR stands for latent regression.

Other Software

Since the higher-order latent structure within IRT can be viewed as a reparameterization of other multidimensional IRT models (Rijmen, 2010), our software review considered any multivariate IRT package which can fit the bifactor (B-IRT) model. There are many currently available packages which estimate multivariate IRT models, such as the bifactor model. Outside of the R environment, SAS provides PROC IRT (Matlock & Paek, 2017), MPLUS (Muthén & Asparouhov, 2013), and IRTPRO (Paek & Han, 2013) to name popular choices, but none of the programs allowed the extension of latent regression.

Inside of the R environment (Team, R Core (2021)), flirt (Jeon et al., 2014) provides a higher-order IRT procedure with latent regression using an expectation-maximization algorithm but the package is only available to Windows users and requires a non-standard compiler which requires a complicated installation. While the mirt package (Chalmbers, 2012) does not provide the higher-order model, the documentation provides the correct parameter constraints

for estimating the higher-order loadings without latent regression. To the best of our knowledge, standard errors are not given for the higher-order factor loadings by either package. Both mirt and flirt use expectation-maximization algorithms for estimation, while hlt uses random-walk Markov chain Monte-Carlo (MCMC). The fully Bayesian estimation method of hlt provides the important advantage of posterior credible intervals and standard errors for all model parameters, plus the extension of latent regression.

Summary

The higher-order IRT model (HO-IRT) and its extension of latent regression (HO-IRT-LR) is the focus of the present article. We provide a Bayesian model and statistical software package for the HO-IRT-LR model which can be extended to more complex problems, such as item level regression or multiple group analysis. Currently, there is no software package which can directly estimate the higher-order latent trait structure described here in the context of IRT and its extension of latent regression. The present article combines these two extensions to IRT with a user-friendly and fast software implementation in the widely used statistical programming environment R and its interface to the C++ programming language, RCPP (Eddelbuettel, 2013).

In this paper, we describe the open-source R package hlt, which includes a set of functions for higher-order descriptive item response theory and an extension with latent regression to conduct explanatory IRT using person-covariates. Additionally, we provide functions to summarize and visualize the results from these models. First, we demonstrate how to install and load the package in R. Second, we give a detailed overview of the core package functionality including how to estimate the model and summarize the results. Third, we describe the statistical model which the software implements. Fourth, we give details about the

MCMC algorithm used for estimation. Finally, we conduct an example analysis using real survey data.

Installation and usage

The R package hlt requires R (version > 3.5.0) and Rcpp (≥ 0.12.0). Pre-build binaries for the current official release of the package are available from CRAN (Comprehensive R Archive Network), but if the user desires to compile the package for himself, then the typical R compiler tools will have to be installed. Instructions can be found on the R Project's website at https://www.r-project.org/nosvn/pandoc/devtools.html.

The latest development version of the package can be found on Github (https://github.com/mkleinsa/hlt). The development version contains updates to the software which are typically a few weeks ahead of the CRAN release.

The hlt package can be installed from CRAN and loaded via the R console with the following commands:

```
R> install.packages("hlt")
R> library(hlt)
```

To install the development version of hlt, install the devtools package (Wickham et al., 2021) and run the following command:

```
R> devtools::install_github("mkleinsa/hlt")
R> library(hlt)
```

See the README of the hlt package on the Github repository page for additional details about package installation. To install the development version of the package, compilation of the C++ code is required at install, so the user's development toolchain must at least include a C++ compiler.

Core functionality

The main model fitting function, `hlt()`, encapsulates all of the package's estimation procedures. The output is either an object of class `hltObj` for a single run or an `hltObjList` for multiple parallel runs. For each object class, R generics are available to `print()`, `plot()`, produce a `summary()` or merge chains (`merge_chains()`) to merge multiple parallel runs.

Help Pages

The functions available to the user of this package each contain detailed documentation in the form of man files. To view the documentation for a given function, run ?FUNCTION_NAME at the R console to view the man page. We also provide example data sets and function calls to demonstrate usage of each function within the *Examples* section of the documentation for the main functions. Run `?hlt` to view the main documentation page and full examples.

Formatting Input Data

The main input to the hlt function, called `x`, is a numeric matrix of item responses from the survey. Data can only be inputted in the wide format, where each row of the matrix contains all responses for one survey respondent, and each column represents the question asked of each respondent. Respondents may not be asked a different number of questions, resulting in a ragged matrix. Also, there may not be missing values in any entries of the data matrix.

For each item in the item response matrix, the responses must be whole numbers where the lowest value is 0 and the highest value is the maximum possible response for the item minus one with no gaps. For example, if a question is asked with 5 possible response values, then the possible values should be lowest = 0,1,2,3,4 = highest. For dichotomous items, use no = 0 and yes = 1.

In addition to the required data matrix `x`, the following arguments to the function are required to fit the model: `id`, a numeric vector indexing first-order latent domain membership for each domain. We index starting from zero, not one. For example, if there are three first-order domains with five questions per domain, then the corresponding `id` vector is `c(0,0,0,0,0,1,1,1,1,1,2,2,2,2,2)`; `iter`, the number of total iterations including burn-in; `burn`, the number of burn-in iterations; `delta`, the random-walk proposal standard deviation; and `type`, the type of model to fit: currently the partial credit model, or PCM (`type = "1p"`), or the generalized partial credit model, or GPCM (`type = "2p"`), are available. For latent regression IRT, the optional `z` argument must be specified. `z` is a standardized numeric matrix of predictors. The dimension of `z` must match the number of rows of `x` and have one column for each regression parameter to be estimated. All columns of the `z` matrix must be numeric. For categorical or factor level of measurement, dummy variable must be coded ($n - 1$ indicator columns for a factor with n levels). For continuous items, the columns should be standardized (mean of zero with standard deviation of one).

In "Illustration of functions with simulated data", we demonstrate the models which the hlt package can fit using simulated data and demonstrate the syntax of the auxiliary functions for viewing the results.

Illustration of functions with simulated data

The hlt package includes a function to simulate data under each of the modeling scenarios. In this section, we demonstrate how to simulate data and estimate the correct higher-order IRT model. For full documentation of the simulation function with examples, run `?hltsim` at the R console.

Simulation Function

The `hltsim` function has arguments for the type of higher-order IRT model (`type`), the sample size (`n`), the number of latent domains (`ntheta`), the true loadings for each latent domain (`lambda`), the domain membership of each latent domain (`id`), the number of levels of each question (`dL`), the number of regression parameters (`nB`), and the true regression parameter values (`beta`).

To simulate the generalized partial credit model without regression, the function call is as follows:

```
R> xdat = hltsim(n = 250, type = "2p", ntheta = 4,
     lambda = c(0.5, 0.8, 0.9, 0.4), id = c(rep(0, 15),
     rep(1, 15), rep(2, 15), rep(3, 15)), dL = 2)
```

In this example, we simulate data for 250 participants from a survey measuring four domains with 15 items per domain. Each response is binary, yes/no (1 or 0). In the following sections, we will demonstrate the other features of `hltsim` in demonstrating the modeling procedures.

The `hltsim` function returns a named list of simulated outputs and arguments including the simulated survey data (`x`), regression design matrix (`z`), parameter settings (`s.lambda`, `s.alpha`, `s.delta`, `s.beta`), and domain I.D. vector (`id`).

Descriptive higher-order IRT

The descriptive higher-order item response theory model has two components: a measurement model which describes the probability of responding correctly to the given survey item; and a factor structure model which describes the relationship between the general factor and domain specific factors.

Generalized Partial Credit Model

We focus on the partial credit model (PCM) and the generalized partial credit model (GPCM) for their flexibility (Muraki, 1992; Masters, 1982). The PCM and

GPCM are flexible enough to account for dichotomous and polytomous response items, covering a wide variety of survey question types. In the dichotomous case, the PCM model is mathematically equivalent to the 1-parameter logistic model and the GPCM is equivalent to the 2-parameter logistic model. Thus, four different types IRT measurement models can be fit with this package, and any number of dichotomous and polytomous item types is possible.

For person i, question j, and domain k, the generalized partial credit model is given by:

$$P\left(Y_{ij} = y|\theta_{ik}, \kappa_j, \alpha_j\right) = \frac{\exp\left(\sum_{l=1}^{y} \nu_j * \theta_{ik} - \kappa_{jl}\right)}{1 + \sum_{w=1}^{n_j} \exp\left(\sum_{l=1}^{w} \nu_j * \theta_{ik} - \kappa_{jl}\right)}$$

Equation 1

Here, $y_{ij} \in \{1, \ldots, n_j\}$ is the observed response for the given person and question for a question with n_j possible response options. The parameter κ_{jl} is the difficulty of responding to item j at level l, ν_j is the discrimination of item j, and θ_{ik} is the person level ability for person i responding to an item belonging to factor domain k.

To improve estimation, the parameterization of Equation 1 was used, where ν_j and κ_{jl} are linearly related. In the parameterization of Muraki (1992), we have $\nu_j(\theta_{ik} - \kappa^*_{jl})$. Thus, the item difficulty of Muraki, κ^*_{jl}, can be obtained by the transformation $\kappa^*_{jl} = \kappa_{jl}/\nu_j$.

For estimability, the difficulty parameter for responding positively in the lowest response category of item j, δ_{j1}, is set to zero. For the partial credit model, we set α_j's to 1. If $Y_{ij} \in \{0,1\}$, then the GPCM is simplifies to the 2-parameter logistic (2-PL) model. Thus, any combination of items with two or more levels can be fit with this likelihood. When with alpha's are set to 1 and $Y_{ij} \in \{0,1\}$, we are left with the 1-parameter logistic (1-PL) model.

Specifying the descriptive HO-IRT model is done with the `type` argument of the `hlt` function. Use `type="1p"` for the PCM (1-PL in the dichotomous case) and `type="2p"` for the GPCM (2-PL in the dichotomous case).

Higher-Order Latent Trait Model

Let θ_{Gi} be the general ability for the ith person and λ_k be the loading for the kth domain. Then, the linear higher-order factor structure is defined as:

$$\theta_{ik} = \lambda_k \theta_{Gi} + \varepsilon_{ik}$$

Equation 2

$$\lambda_k \sim TruncN(0{,}10, -10{,}10)$$

Equation 3

$$\varepsilon_{ik} \sim N(0{,}1)$$

Equation 4

When only two domains are specified, the model is no longer identifiable (de la Torre, 2009). When this is the case, the λ_k's are given an equality constraint, i.e., $\lambda_1 = \lambda_2$.

The following R code fits the HO-IRT model to the simulated data set:

```
R> mod1 = hlt(x = xdat$x, id = xdat$id, iter = 12e5,
     burn = 6e5, delta = 0.023)
```

Explanatory higher-order IRT

This descriptive IRT model can be extended with person and item level covariates, called explanatory IRT. The hlt package implements the person covariate approach to explanatory IRT.

Let $\{\beta_1, \ldots, \beta_p\}$ be the set of regression coefficients for p explanatory variables in the latent regression model for general ability. Then, the linear latent regression model of general ability is given by,

$$\theta_{Gik} = \beta_1 x_{1i} + \cdots + \beta_p x_{pi} + \xi_{ik}$$

Equation 5

$$\xi_{ik} \sim N(0{,}1)$$

Equation 6

To adapt the previous simulation to include latent regression, the function call is as follows:

```
R> xdat = hltsim(n = 250, type = "2p", ntheta = 4,
     lambda = c(0.5, 0.8, 0.9, 0.4), id = c(rep(0, 15),
     rep(1, 15), rep(2, 15), rep(3, 15)), dL = 2,
     beta = c(0.5, -0.7), nB = 2)
```

Note that two arguments were added: `beta = c(0.5, -0.7)`, which sets two new regression coefficients, and `nB = 2`, which signifies the number of regression coefficients set. The corresponding hlt call to fit the model is:

```
mod2 = hlt(x = xdat$x, id = xdat$id, z = xdat$z,
     iter = 12e5, burn = 6e5, delta = 0.023, nchains = 1)
```

Model Tuning

The random walk Metropolis-Hastings algorithm requires one tuning parameter, the root proposal variance (option `delta` in `hlt()`), which is the standard deviation of the normally distributed proposal. If alpha is too large, then the proposal will reject too often, leading to not enough exploration of the posterior density. If alpha is too small, then too many draws will be accepted, and the proposal density will not be explored. The optimal acceptance rate for multidimensional problems like this one is 0.234 (Rosenthal, 2011).

To save time, it is recommended to run a sequence of models, beginning with few draws, and incrementally increasing the number of draws to monitor the acceptance rate. Once the acceptance rate approaches 0.234, run one long chain until convergence is achieved. This step is often followed by another multiple chain run with at least three chains to confirm agreement between independent chains. For example, to find `delta = 0.023` in the previous example, we set `iter` to `1e3`, `1e4`, and `1e5`. By `1e5` iterations, a delta of `0.023` gave an acceptance rate of `0.214`, which is acceptable.

Important Options

We also provide the start argument of hlt for user-specified starting values. `start` is given as a named list, which can be viewed on the `?hlt` man page. We also include a function (`get_hlt_start(x, nchains)`) to use the results of the previous run as starting values.

For multiple chain MCMC, we include the `nchains` argument of `hlt` to specify n parallel chains to run. If nchains > 1, then `get_hlt_start` must match the length of the starting values list, e.g., start = `get_hlt_start(mod1, nchains = 3)`.

Summary, Plot, and Merge

The hlt package implements the `hltObj` and `hltObjList` classes which impalement the R S3 generic functions plot and summary, so that R users can inspect the output of `hlt` in the same way as the core R functions. This section overviews how to interact with the objects which the package outputs.

Print

Calling `print()` on an hlt object results in a brief model fit summary including the number of domains evaluated, total number of model parameters, model type and other input settings, acceptance rate, factor loadings and regression coefficients. For more detailed summaries, the `summary()` function is needed.

Summary

Model estimates are made available by calling `summary()` on an hlt model object. Summary takes two primary arguments: 1) an `object` of class `hltObj` in the case of a single chain, or an object of class `hltObjList` in the case of a multiple chain object; 2) `param`, the name of the parameters to summarize. Summary provides posterior mean, standard deviation, and 2.5%, 50%, and 97.5%

quantiles for each of the test level parameters. Posterior estimates of ability parameters show only the mean and standard deviation since there are too many individual level parameters to save every draw.

To display summaries of all parameter estimates at once (except for the ability estimates), use `param = "all"`.

Extract latent factor loadings (`param = "lambda"`), latent regression coefficients (`param = "beta"`), difficulty (`param = "delta"`) and discrimination parameters (`param = "alpha"`), and abilities (`param = "theta"`) from each dimension by also specifying the required dimension, e.g., for the first dimension, call `summary(mod2, param = "theta", dimension = 1)`. If a survey has n domains, then the dimension argument ranges from 1 to $n + 1$, where $n + 1$ represents the higher-order domain. For example, if a survey has 4 domains, then the dimension arguments takes the values $1, 2, 3, 4$, or 5 for the higher-order ability parameters.

If the user prefers to instead obtain the draws to use elsewhere, the hltObj object contains a named attribute post, which is the matrix of posterior draws after burn-in.

Plot

Similarly, model diagnostic and summary plots are available by calling `plot()` on an hlt model object. Plot requires the arguments `x`, `param`, and `type` for MCMC traceplots, or `x`, `item`, and `type` for item characteristic curves, item information curves, and the total information curve. Unlike the summary function, if a traceplot for a parameter is desired, the user must explicitly give the full name of the model parameter. For a full list of parameter names from an hltObj object, use `summary(mod, param = "all")` or `colnames(mod$post)`. For

example, to retrieve the traceplot for the second level of the fifth difficulty parameter, use:

```
R> plot(mod, param = "d5_2", type = "trace")
```

Traceplots are not available for the ability, or theta, parameters since saving each draw of the individual level parameters uses too much memory. Thus, we only save means and standard deviations of the ability parameters.

One can also obtain traceplots when multiple parallel chains are run by calling `plot` on an hltObjList object.

Also with `plot`, item characteristic, item information, and total information curves can be obtained. Item I.D's can range from 1 to the number of items asked in the survey. For example, to produce the item characteristic (`"icc"`), item information (`"iic"`) , and total information (`"tic"`) curves, use the following syntax:

```
R> plot(mod, item = 5, type = "icc")
R> plot(mod, item = 5, type = "iic")
R> plot(mod, type = "tic")
```

Merge

A merge function called `merge_chains` is available to merge multiple chain objects (nchains > 1) into one object. This makes running summary and plot on the combined results more convenient.

Implementation

Given the complexity of the model, we used the Bayesian technique of MCMC simulation to obtain estimates of model parameters. Specifically, we implemented the random walk Metropolis-Hastings MCMC algorithm to estimate the full joint posterior distribution. The sampler was implemented in C++ for the comparative

speed advantages of C++ over R. We used recourse to another language because pre-compiled languages like C++ and Fortran are more efficient at repeating the same task millions of times, and this makes a significant difference if the model has thousands of parameters, as our models do.

To implement the random walk Metropolis-Hastings algorithm, one needs only to describe the log of the joint posterior distribution for the desired model. In this section, we describe the joint posterior distribution in terms of the joint likelihood and prior distributions.

Let $\boldsymbol{x}$ be an N by J matrix of item responses where each element represents the n^{th} ($n = 1, \dots, N$) participant's response to the j^{th} ($j = 1, \dots, J$) question. Let the j^{th} vector of item response, x_j, take the values 0 to max $(x_j) - 1$. Also, let $\boldsymbol{z}$ be an N by P ($p = 1, \dots, P$) matrix of person covariates. The parameters of interest are the length J item discriminations ν, the J vectors of item difficulties κ_j, the factor loadings λ, the domain specific ability vectors θ_k for $k = 1, \dots, K$, the general ability vector θ_G, and the P latent regression slope parameters β.

We seek to do inference on the joint posterior distribution,

$$P(\nu, \kappa_1, \dots, \kappa_J, \theta_G, \theta_1, \dots, \theta_K, \lambda, \beta | \boldsymbol{x}, \boldsymbol{z})$$

Equation 7

Using Bayes theorem, the joint posterior distribution of the set of parameters $\{\nu, \kappa_1, \dots, \kappa_J, \theta_G, \theta_1, \dots, \theta_K, \lambda, \beta\}$ given data $\boldsymbol{x}$ and $\boldsymbol{z}$ is:

$$P\left(\nu, \kappa_1, \dots, \kappa_J, \theta_G, \theta_1, \dots, \theta_K, \lambda, \beta \middle| \boldsymbol{x}, \boldsymbol{z}\right) \propto P\left(\boldsymbol{x}, \boldsymbol{z} \middle| \nu, \kappa_1, \dots, \kappa_J, \theta_G, \theta_1, \dots, \theta_K, \lambda, \beta\right) * P(\nu) * P(\kappa_1) * \cdots * P(\kappa_J) * P(\theta_G) * P(\theta_1) * \cdots * P(\theta_K) * P(\lambda) * P(\beta)$$

Equation 8

Given that we can specify the right-hand side of (5), what is left is to specify the joint sampling distribution and the joint posterior distribution.

The joint sampling distribution is given by:

$$P\left(\boldsymbol{x}, \boldsymbol{z} \middle| \nu, \kappa_1, \ldots, \kappa_J, \theta_G, \theta_1, \ldots, \theta_K, \lambda, \beta\right) = \prod_{i=1}^{N} \prod_{j=1}^{J} \prod_{k=1}^{K} \frac{\exp\left(\sum_{l=1}^{y} \nu_j * \theta_{ik} - \kappa_{jl}\right)}{1 + \sum_{w=1}^{n_j} \exp\left(\sum_{l=1}^{w} \nu_j * \theta_{ik} - \kappa_{jl}\right)}$$

Equation 9

The prior distributions for the set of model parameters $\{\nu, \kappa_1, \ldots, \kappa_J, \theta_G, \theta_1, \ldots, \theta_K, \lambda, \beta\}$ are given by (10) – (15):

$$\nu_j \sim TruncN(0,2,0,10)$$

Equation 10

Here, $TruncN(\mu, \sigma, a, b)$ is the truncated normal distribution and $N(\mu, \sigma)$ is the normal distribution with mean μ and standard deviation σ truncated at lower bound a and upper bound b.

$$\kappa_{jl} \sim TruncN(0,10)$$

Equation 11

$$\theta_{ki} \sim N(\lambda_k \theta_{Gi}, 1)$$

Equation 12

$$\lambda_k \sim TruncN(0,10,-10,10)$$

Equation 13

$$\theta_{Gi} \sim N\left(\beta_1 z_{i1} + \cdots + \beta_p z_{ip}, 1\right)$$

Equation 14

$$\beta_p \sim N(0,10)$$

Equation 15

In the case where the model is fitted without latent regression, (14) becomes $\theta_{Gi} \sim N(0,1)$. The prior distributions were chosen to conform to prior studies and hyperpriors were chosen to be vague unless model constraints required strong prior information. For example, in (10), (12), and (14), prior variance of the factors and their loadings were constrained for estimability.

Random Walk Metropolis-Hastings Algorithm

In general, the goal of Markov chain Monte Carlo (MCMC) is to sample from a target random variable with probability distribution $P(\cdot)$ by generating a Markov chain with stationary distribution $P(\cdot)$ (Sherlock et al., 2010). In our case, the probability distribution of interest is the posterior distribution given in (7).

One popular choice of MCMC algorithm is random walk Metropolis-Hastings (Metropolis et al., 1953; Hastings, 1970). In this algorithm, the log posterior density is evaluated at its current position, x, and a proposed position, $x + x^*$, sampled from the Normal probability density. When the log density of the proposal is greater than that of its current position, the draw is excepted. Otherwise, the draw is accepted with probability $\min(1, \pi(x + x^*)/\pi(x))$. For a good review of random walk Metropolis-Hastings, see Sherlock et al., 2010. Presented below are the details of the random walk Metropolis-Hastings algorithm used to sample from the joint posterior distribution of the HO-IRT-LR model:

1. Set initial values for the parameters $\{\nu, \kappa_1, \ldots, \kappa_J, \theta_G, \theta_1, \ldots, \theta_K, \lambda, \beta\}$. In the package, there are two ways of setting initial values using the `start` argument of the hlt function. If start is not specified, then random starting values are chosen. If start is specified, then the user specified starting values are used. The syntax for specifying starting values is given by `start = list(list(lambda = c(), theta = c(), delta = c(), alpha = c(), beta = c()))`. Here, we provide a list containing a list of named starting values for each parameter and each chain. The length of each vector should match the number of parameters to be estimated. Alternately, use `get_hlt_start()`.
2. Let t iterate over the total number of draws and set $t = 1$. Then for each iteration, do:
    a. Generate a random variate x^* from $N(x, \sigma)$. The standard deviation of the proposal distribution, σ, is specified by the `delta` parameter of `hlt()`.
    b. Calculate the acceptance ratio: $a = \min(1, \pi(x^*)/\pi(x))$
    c. Acceptance/rejection step:

i. Generate u from $Unif(0, 1)$

ii. If $u \leq a$, set $x_{t+1} = x^*$, else set $x_{t+1} = x$

Note that during model tuning, the delta parameter is adjusted until an acceptance rate of roughly 0.234 is achieved. The user can retrieve the acceptance rate from an hlt object called mod with `mod$accept.rate` or `print(mod)`.

Example data analysis: ASTI

Data set

The adult self-transcendence inventory, or ASTI, was originally created to assess the construct of wisdom through self-transcendence (Levenson et al., 2005). Recently, Koller et al. (2017) re-analyzed 24 ASTI items measuring self-transcendence over five dimensions. The five domains of the Koller ASTI survey were: 1) self-knowledge and integration (four items); 2) peace of mind (four items); 3) non-attachment (four items); 4) self-transcendence (seven items); 5) and presence in the here-and-now and growth (six items). A primary goal of the study was to determine if the ASTI formed a unidimensional scale. However, since the five dimension each form their own theoretical domain of wisdom, a unidimensional scale may be an oversimplification of the problem. This is further supported by the fact that the authors found that a multidimensional IRT (M-IRT) model with five dimensions fit the data better than the univariate model. Higher-order IRT (HO-IRT) tests the correct hypothesis if what is sought is a single general dimension governing the hypothesized five domains of wisdom.

In this example, we demonstrate the process of fitting the generalized partial credit HO-IRT (descriptive) model to the ASTI data and we assess the effect of gender (male/female) and student status (student/non-student) on the higher-order transcendence dimension using the HO-IRT-LR (explanatory) model. For a

graphical representation of the HO-IRT-LR model of the ASTI data, refer to Figure 1.

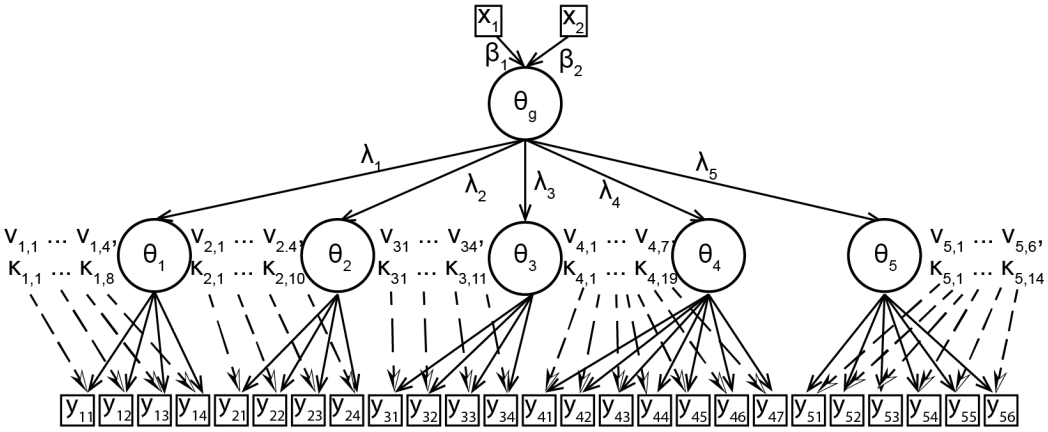

Figure 1. Higher-Order IRT Latent Regression Diagram for ASTI Data. Rectangles represent observed data, circles represent latent data, and the other parameters are left not circled. The notation used here matches Eq. 1.

Data preparation

The publicly available data set comes from the R package MPsychoR on CRAN (Mair 2018), but it is also available through the hlt package with:

```
R> library(hlt)
R> data("asti")
```

Preparing the Input

The model fitting procedure occurs in two stages: first we do a single chain run of the algorithm to tune the acceptance rate and to monitor convergence. Once convergence is achieved, we use the starting values from the previous solution and run three parallel chains. Finally, the results are summarized. The hlt model fitting function requires that survey responses from a given question i are integers ranging from 0 to $n - 1$. However, the ASTI data are coded from $1, \ldots, n$, so the responses are adjusted with:

```
R> x = asti[, 1:25] - 1
```

The `id` vector for the ASTI data requires 5 unique integer levels (one level for each domain of the survey) again coded from 0 to $k - 1$:

```
R> id = c(rep(0, 4), rep(1, 4), rep(2, 4), rep(3, 7), rep(4,
          6))
```

Note that the values of `id` must be an ordered sequence and the domain cannot overlap, e.g., `c(0,0,0,1,0,0,1,1)` is invalid and should rather be `c(0,0,0,0,0,1,1,1)`.

Lastly, in preparation for the latent regression analysis, we create a matrix of dummy variables for each variable included:

```
R> z = asti[, 26:27]
R> z[, 1] = (z[, 1] == "students") * 1
R> z[, 2] = (z[, 2] == "male") * 1
```

Notice that this example includes two factor which each have two levels, thus $2 - 1$ dummy variables are required of each variable. For continuous variables, a singe numeric vector is required, but that each numeric variable also be standardized.

Model Fitting

We first conduct a descriptive analysis to assess the fit of the HO-IRT model in terms of the item parameters using the GPCM. We then fit the HO-IRT-LR model to estimate the latent regression coefficients. While the analysis did not need to proceed in stages, it was done to show the software's capacity for both facets if IRT.

Descriptive Analysis

In the descriptive case, we seek to describe the properties of each scale within the context of a higher-order latent trait structure. These properties are characterized by the item parameters of discrimination and difficulty. Without prior knowledge about an acceptable proposal variance or starting values that are reasonably close

to the posterior mode, we first need to tune the sampler and then sample until the posterior mode stops changing. Then, for the final run, we provide a set of starting values and run n parallel chains, check the trace plots, merge the chains, and summarize the results.

We first ran the model at `1e2`, `1e3`, `1e4`, respective numbers of iterations, tuning the proposal variance parameter `delta`. Once an acceptance rate of approximately 0.234 was achieved, we ran one chain until the posterior mode was reached. Then, we ran three chains using the single chain starting values. The model fitting script is as follows:

```
R> asti_gpc = hlt(x = x, id = id, iter = 2.5e6, burn = 2e6,
        delta = 0.01, type = "2p")

Higher-order item response theory model with 5 first-order
domains
 Total number of parameters: 6879
 Model type (1p = "Partial Credit model"; 2p = "Generalized
Partial Credit Model"): 2p
 iterations: 2500000; burn-in: 2e+06
 Proposal standard deviation: 0.01
 Acceptance rate: 0.225

Loadings:  1.276 1.537 0.43 0.285 1.712

R> asti_gpc_m = hlt(x = x, id = id, iter = 2e5, burn = 1e5,
        delta = 0.01, type = "2p", nchains = 3,
        start = get_hlt_start(asti_gpc, nchains = 3))
```

It is essential that the MCMC chains are monitored for convergence. Once a model is fitted trace plots for the parameters should be checked individually with:

```
R> plot(asti_gpc_m, param = "lambda4", type = "trace")
```

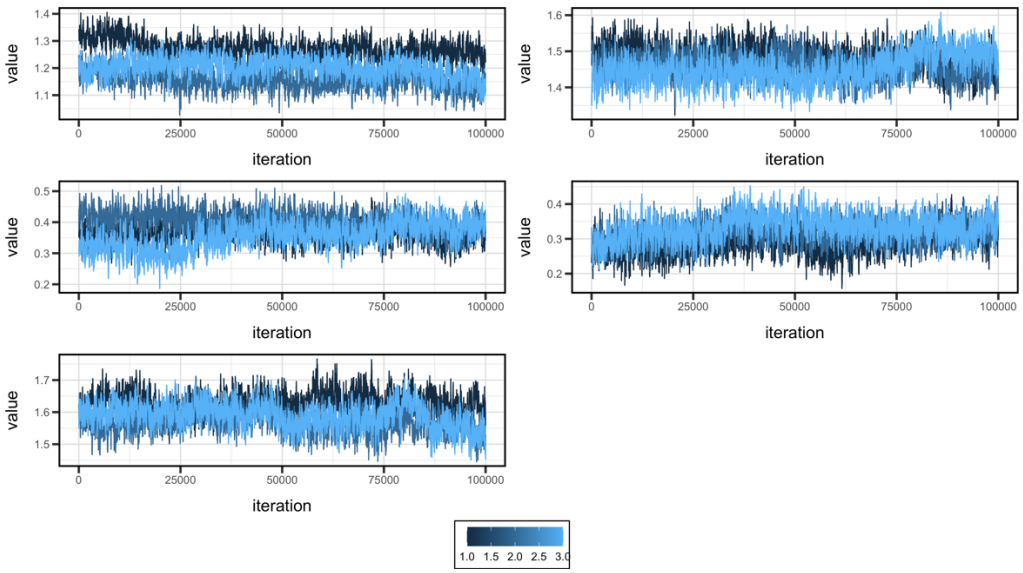

Figure 2. Traceplots of the five latent factor loadings. Traces are shown for post burn-in iterations for lambda1 (top left), lambda2 (top right), lambda3 (middle left), lambda4 (middle right), and lambda5 (bottom left).

Figure 2 displays the MCMC chains for four selected parameters. For a complete set of parameter names in the model, use `colnames(asti_gpc$post)`. Next, we show how to view the item level parameter estimates using the model summary function after merging the multiple chains:

```
R> asti_merg = merge_chains(asti_gpc_m)
R> summary(asti_merg, param = "alpha")
R> summary(asti_merg, param = "delta")
```

The item characteristic, item information, and total information curves are produced with the following code (see Figure 3, below):

```
R> plot(asti_merg, item = 1, type = "icc", min = -6,
     max = 6)
R> plot(asti_merg, item = 1, type = "iic", min = -6,
     max = 6)
R> plot(asti_merg, item = 1, type = "tic", min = -6,
     max = 6)
```

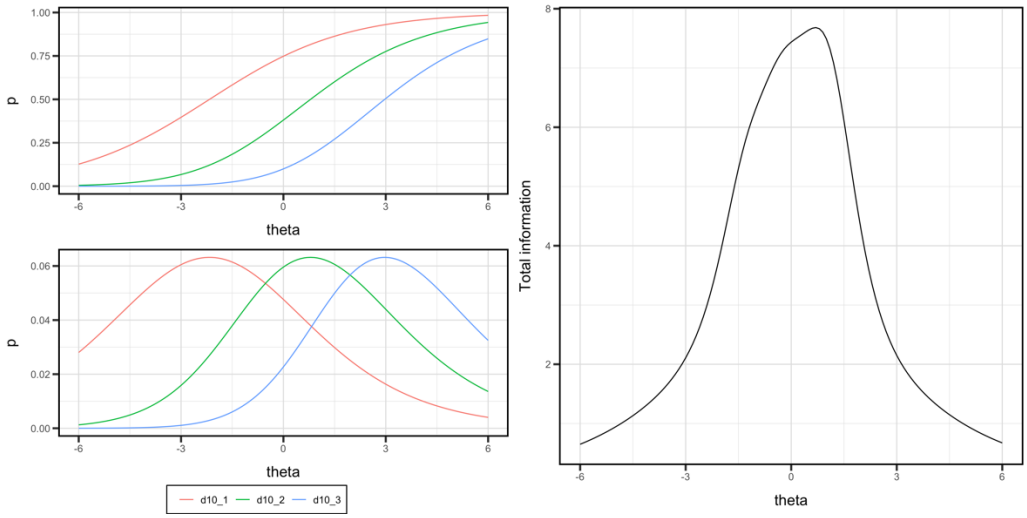

Figure 3. Item characteristic curve for item 1 (top left), item information curve for item 1 (bottom left), and total information curve (right).

The relationship between latent dimension can be characterized by the raw factor loadings, correlations between the general factor and each first-order domain, and the correlations among the first-order domains themselves, as shown below and in Figure 4:

```
R> summary(asti_merg, param = "lambda")

          mean    se  2.5%   50% 97.5%
lambda1 1.209 0.059 1.105 1.207 1.326
lambda2 1.464 0.038 1.389 1.464 1.540
lambda3 0.376 0.042 0.287 0.378 0.453
lambda4 0.313 0.039 0.236 0.313 0.387
lambda5 1.595 0.044 1.511 1.595 1.682

R> summary(asti_merg, param = "correlation")

       theta1 theta2 theta3 theta4 theta5 theta6
theta1  1.000  0.787  0.406  0.281  0.824  0.895
theta2  0.787  1.000  0.477  0.344  0.853  0.937
theta3  0.406  0.477  1.000  0.236  0.427  0.510
theta4  0.281  0.344  0.236  1.000  0.367  0.402
theta5  0.824  0.853  0.427  0.367  1.000  0.951
theta6  0.895  0.937  0.510  0.402  0.951  1.000
```

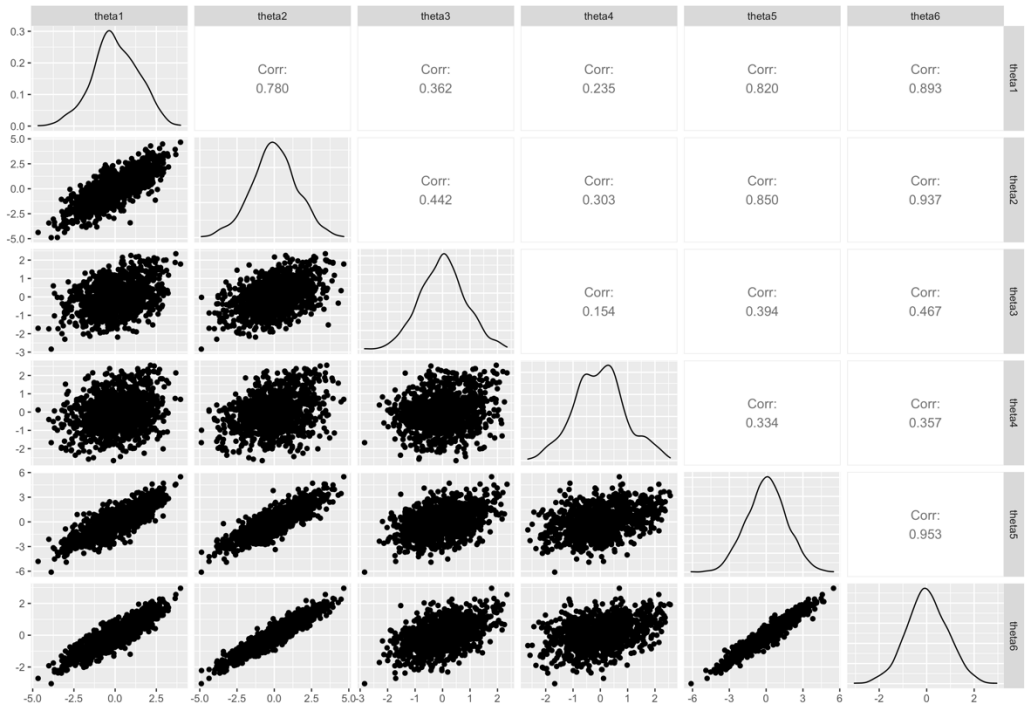

Figure 4. Correlogram showing the bivariate correlations (upper triangle), scatter plots (lower triangle), and density plots (diagonal) for each first-order domain (theta1 through theta5) and the higher-order domain (theta6).

The ability estimates are extracted with summary for each of the first-order domains and the higher-order domain. Since there are five first-order domains, dimension = 1,…,5 will give each respective summary, and dimension = 6 givens the higher order abilities:

```
R> # First dimension
R> summary(asti_merg, param = "theta", dimension = 1)
R> # higher-order dimension
R> summary(asti_merg, param = "theta", dimension = 6)
```

Explanatory Analysis

In the explanatory case, the person characteristics of gender and student status are used to explain heterogeneity in the ability of persons to answer questions correctly. This analysis is much simpler given the work we did in the descriptive analysis, which provided good starting values and proposal variance. Now, we repeat the previous analysis, but this time by also specifying a latent regression model:

```
R> asti_gpc_reg = hlt(x = x, id = id, z = z, iter = 2.5e6,
     burn = 2e6, delta = 0.01, type = "2p",
```

```
     start = get_hlt_start(asti_merg, nchains = 1),
     nchains = 1)
R> asti_gpc_reg
Higher-order item response theory model with 5 first-order
domains
 Total number of parameters: 6881
 Model type (1p = "Partial Credit model"; 2p = "Generalized
Partial Credit Model"): 2p
 iterations: 2500000; burn-in: 2e+06
 Proposal standard deviation: 0.01
 Acceptance rate: 0.205

Loadings:  1.443 1.517 0.49 0.244 1.579
Latent regression beta estimates:  -0.118 -0.058

R> asti_gpc_reg_m = hlt(x = x, id = id, z = z, iter = 1e6,
     burn = 5e5, delta = 0.01, type = "2p",
     start = get_hlt_start(asti_gpc_reg, nchains = 3),
     nchains = 3)
```

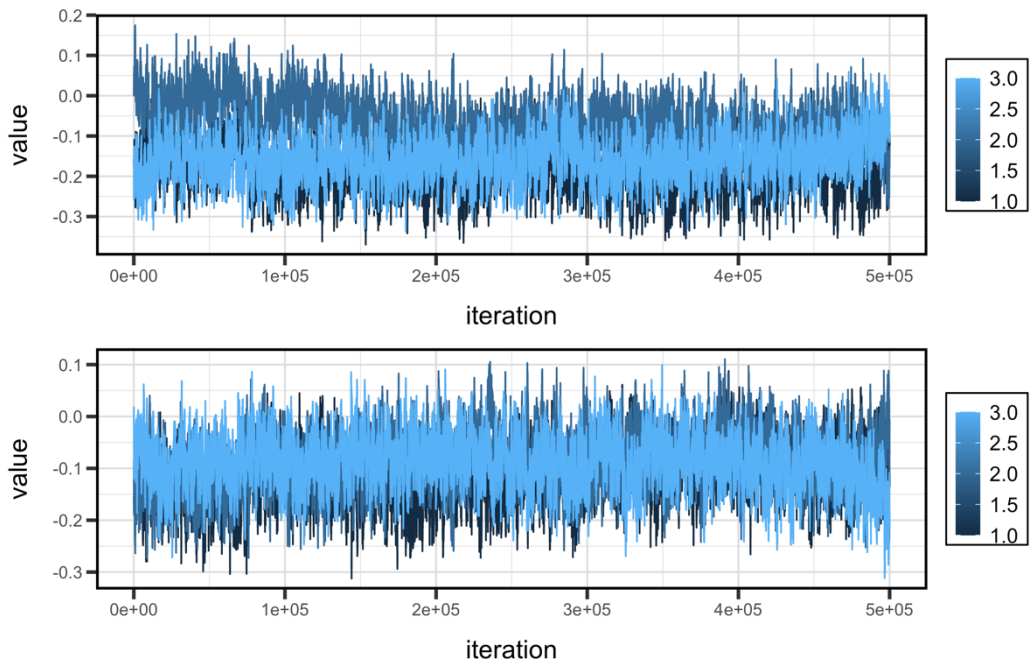

Figure 5. Traceplots of latent regression coefficients. Traces are shown for post burn-in iterations for beta1 (top) and beta2 (bottom). Beta1 represents "students" and beta2 represents "males".

We then merge the chains into one summary and inspect the results:

```
R> asti_gpc_reg_mm = merge_chains(asti_gpc_reg_m)
R> summary(asti_gpc_reg_mm, param = "beta")

        mean    se   2.5%    50% 97.5%
beta1 -0.137 0.076 -0.272 -0.143 0.023
beta2 -0.096 0.053 -0.201 -0.095 0.008
```

Final Remarks

While the hlt package can detect heterogeneity at the level of ability, item, or test, level heterogeneity through differential item functioning and multiple group analysis have not been implemented. Given the importance of detecting heterogeneity at the level of the person and the test, item level regression is our next intended future development. Additionally, though the GPCM covers a variety of item types, future developments will involve implementing other measurement models like the graded response, three-parameter, or even four-parameter models.

Our implementation of the random walk Metropolis algorithm requires the user to tune the variance of the Normal proposal density themselves. Since the proposal variance is essential to algorithm convergence, this puts a higher burden on the user to ensure accurate results. One solution is to implement an adaptive random walk algorithm which tunes the proposal variance as sampling occurs (Rosenthal 2011). Lastly, a set of model fit statistics, such as the deviance information criterion, and diagnostic plots of the posterior predictive checks would improve comparison and evaluation of competing models.

References

Cai, L., Choi, K., Hansen, M., & Harrell, L. (2016). Item response theory. Annual Review of Statistics and Its Application, 3, 297-321.

Chalmers, R. P. (2012). mirt: A multidimensional item response theory package for the R environment. Journal of statistical Software, 48, 1-29.

Costa Jr, P. T., & McCrae, R. R. (1995). Domains and facets: Hierarchical personality assessment using the Revised NEO Personality Inventory. Journal of personality assessment, 64(1), 21-50.

De Boeck, P., & Wilson, M. (Eds.). (2004). Explanatory item response models: A generalized linear and nonlinear approach (Vol. 10, pp. 978-1). New York: Springer.

de la Torre, J., & Song, H. (2009). Simultaneous estimation of overall and domain abilities: A higher-order IRT model approach. Applied Psychological Measurement, 33(8), 620-639.

Eddelbuettel, D. (2013). Seamless R and C++ integration with Rcpp. New York: Springer.

Golden-Kreutz, D. M., Browne, M. W., Frierson, G. M., & Andersen, B. L. (2004). Assessing stress in cancer patients: a second-order factor analysis model for the Perceived Stress Scale. Assessment, 11(3), 216-223.

Gotay, C. C., Blaine, D., Haynes, S. N., Holup, J., & Pagano, I. S. (2002). Assessment of quality of life in a multicultural cancer patient population. Psychological Assessment, 14(4), 439.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications.

Hsieh, H-F., Mistry R., Kleinsasser, M., Puntambekar, N., Gupta, P., Raghunathan, T., McCarthy, W., Córdova, D., Maharjan, G., Desai, M. B., Narake, S., Pednekar, M.. (2022). Family Functioning within the Context of Families with Adolescent Children in Urban India. Family Process.

Huang, H. Y., Wang, W. C., Chen, P. H., & Su, C. M. (2013). Higher-order item response models for hierarchical latent traits. Applied Psychological Measurement, 37(8), 619-637.

Jeon, M., Rijmen, F., & Rabe-Hesketh, S. (2014). Flexible item response theory modeling with FLIRT. Applied Psychological Measurement, 38(5), 404-405.

Koller, I., Levenson, M. R., & Glück, J. (2017). What do you think you are measuring? A mixed-methods procedure for assessing the content validity of test items and theory-based scaling. Frontiers in psychology, 126

Levenson, M. R., Jennings, P. A., Aldwin, C. M., & Shiraishi, R. W. (2005). Self-transcendence: Conceptualization and measurement. The International Journal of Aging and Human Development, 60(2), 127-143.

Mair, P. (2018). Modern psychometrics with R. Cham: Springer International Publishing.

Marsh, H. W., & Hocevar, D. (1985). Application of confirmatory factor analysis to the study of self-concept: First-and higher order factor models and their invariance across groups. Psychological bulletin, 97(3), 562.

Masters, G. N. (1982). A Rasch model for partial credit scoring. Psychometrika, 47(2), 149-174.

Matlock Cole, K., & Paek, I. (2017). PROC IRT: A SAS procedure for item response theory. Applied psychological measurement, 41(4), 311-320.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. The journal of chemical physics, 21(6), 1087-1092.

Muraki, E. (1992). A generalized partial credit model: Application of an EM algorithm. ETS Research Report Series, 1992(1), i-30.

Muthén, B., & Asparouhov, T. (2013). Item response modeling in Mplus: a multi-dimensional, multi-level, and multi-time point example. Handbook of item response theory: Models, statistical tools, and applications, 1-29.

Paek, I., & Han, K. T. (2013). IRTPRO 2.1 for Windows (item response theory for patient-reported outcomes). Applied Psychological Measurement, 37(3), 242-252.

Rijmen, F. (2010). Formal relations and an empirical comparison among the bi-factor, the testlet, and a second-order multidimensional IRT model. Journal of Educational Measurement, 47(3), 361-372.

Rosenthal, J. S. (2011). Optimal proposal distributions and adaptive MCMC. Handbook of Markov Chain Monte Carlo, 4(10.1201).

Sherlock, C., Fearnhead, P., & Roberts, G. O. (2010). The random walk Metropolis: linking theory and practice through a case study. Statistical Science, 25(2), 172-190.

Team, R. C. (2021). R: A language and environment for statistical computing.

Turner, R., & Adams, R. J. (2007). The programme for international student assessment: An overview. Journal of applied measurement, 8(3), 237.

Wickham, H., Hester, J., Chang, W., & Bryan, J. (2021). devtools: Tools to Make Developing R Packages Easier. R package version 2.4. 3.