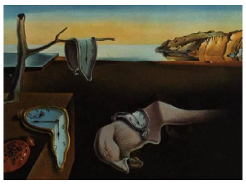


Journal of Statistical Software

May 2012, Volume 48, Issue 6. *http://www.jstatsoft.org/*

mirt: A Multidimensional Item Response Theory Package for the R Environment

R. Philip Chalmers
York University

Abstract

Item response theory (IRT) is widely used in assessment and evaluation research to explain how participants respond to item level stimuli. Several R packages can be used to estimate the parameters in various IRT models, the most flexible being the **ltm** (Rizopoulos 2006), **eRm** (Mair and Hatzinger 2007), and **MCMCpack** (Martin, Quinn, and Park 2011) packages. However these packages have limitations in that **ltm** and **eRm** can only analyze unidimensional IRT models effectively and the exploratory multidimensional extensions available in **MCMCpack** requires prior understanding of Bayesian estimation convergence diagnostics and are computationally intensive. Most importantly, multidimensional confirmatory item factor analysis methods have not been implemented in any R package.

The **mirt** package was created for estimating multidimensional item response theory parameters for exploratory and confirmatory models by using maximum-likelihood methods. The Gauss-Hermite quadrature method used in traditional EM estimation (e.g., Bock and Aitkin 1981) is presented for exploratory item response models as well as for confirmatory bifactor models (Gibbons and Hedeker 1992). Exploratory and confirmatory models are estimated by a stochastic algorithm described by Cai (2010a,b). Various program comparisons are presented and future directions for the package are discussed.

Keywords: multidimensional IRT, model estimation, exploratory item factor analysis, confirmatory item factor analysis, bifactor, R.

1. Introduction

Item response theory (IRT) is widely used in educational and psychological research to model how participants respond to test items in isolation and in bundles (Thissen and Wainer 2001). It is a general framework for specifying the functional relationship between a respondent's underlying latent trait level (i.e., commonly known as 'ability' in educational testing, or 'factor score' in the factor analysis tradition[1]) and an item level stimulus. IRT methodology attempts to model individual response patterns by specifying how the underlying latent traits interact with the item's characteristics — such as an item's easiness or discrimination ability — to form an expected probability of the response pattern. As such, a major goal of IRT is to separate the item parameters and population sampled characteristics from manifest data so that both may be understood and studied separately. This parameter separation often requires advanced numerical analysis techniques for effective estimation and can become computationally burdensome as the model complexity increases.

The simplest and most popular IRT models are those that specify a single (i.e., unidimensional) latent trait. Unidimensional IRT models have been predominant across social science and educational research mainly because of historical traditions, but also because multidimensional parameter estimation procedures were not fully developed or studied (Baker and Kim 2004; Reckase 2009). While unidimensional models are often simpler and can have various interesting and important measurement properties (e.g., Rasch models), many psychological constructs are unavoidably multidimensional in nature. For instance, unobservable constructs might be understood as a combination of sub-scale components nested within — or along-side — a more general construct, or as compensatory or noncompensatory factors that combine to influence the item response probabilities. A major impediment when deciding to utilize these models has been that the estimation of the item parameters in higher dimensional space (due to increasing the number of factors) is computationally difficult for standard numerical integration techniques. However, with recent advances in estimation theory, coupled with the advances in computational power of personal computers, multidimensional IRT research is finally beginning to blossom as a feasible statistical analysis methodology (Edwards 2010; Reckase 2009; Wirth and Edwards 2007).

Several R (R Development Core Team 2012) packages can be used to fit IRT models, such as: the **ltm** package (Rizopoulos 2006), which can handle the Rasch, general latent trait, three-parameter logistic, and graded response models; the **eRm** package (Mair and Hatzinger 2007), which can fit the rating scale and partial credit models; and the **MCMCpack** package (Martin *et al.* 2011), which can estimate k-dimensional unconstrained two-parameter item response models (normal, heteroscedastic, and robust estimation) using a Markov chain Monte Carlo (MCMC) approach. While useful in their own right, these packages have limitations in that **ltm** and **eRm** can only analyze unidimensional IRT models effectively while the multidimensional extensions available in **MCMCpack** require prior understanding of Bayesian estimation diagnostics, are computationally demanding, can require a large amount of memory storage, and are only available for dichotomous item response sets.

2. Item response models

Item response models typically follow a monotonically increasing probability form with respect to the underlying latent traits. Two well known and commonly used logistic response models for dichotomous and polytomous item responses are the Birnbaum (1968) three-parameter model (3PL; which can be reduced to a 1PL or 2PL model) and the Samejima (1969) ordinal response model, respectively. Although first introduced as unidimensional modeling functions, both models can readily generalize to incorporate more than one factor. Let $i = 1, \dots, N$

[1]The terms 'traits' and 'factors' are used interchangeably throughout.

represent the distinct participants, $j = 1, \ldots, n$ the test items, and suppose that there are m latent factors $\boldsymbol{\theta}_i = (\theta_{i1}, \ldots, \theta_{im})$ with associated item slopes $\boldsymbol{\alpha}_j = (\alpha_1, \ldots, \alpha_m)$. For the multidimensional 3PL model, the probability of answering a dichotomous item correctly is

$$\Phi(x_{ij} = 1|\boldsymbol{\theta}_i, \boldsymbol{\alpha}_j, d_j, \gamma_j) = \gamma_j + \frac{(1-\gamma_j)}{1+\exp\left[-D(\boldsymbol{\alpha}_j^\top \boldsymbol{\theta}_i + d_j)\right]} \tag{1}$$

where d_j is the item intercept, γ_j is the so-called 'guessing' parameter, and D is a scaling adjustment (usually 1.702) used to make the logistic metric more closely correspond to the traditional normal ogive metric (Reckase 2009).

For Samejima's (1969) multidimensional ordinal response model, suppose that there are C_j unique categories for item j, with intercepts $\mathbf{d}_j = (d_1, \ldots, d_{(C_j-1)})$. Here we define the boundary of response probabilities as

$$\begin{aligned}
\Phi(x_{ij} \geq 0|\boldsymbol{\theta}_i, \boldsymbol{\alpha}_j, \mathbf{d}_j) &= 1, \\
\Phi(x_{ij} \geq 1|\boldsymbol{\theta}_i, \boldsymbol{\alpha}_j, \mathbf{d}_j) &= \frac{1}{1+\exp\left[-D(\boldsymbol{\alpha}_j^\top \boldsymbol{\theta}_i + d_1)\right]}, \\
\Phi(x_{ij} \geq 2|\boldsymbol{\theta}_i, \boldsymbol{\alpha}_j, \mathbf{d}_j) &= \frac{1}{1+\exp\left[-D(\boldsymbol{\alpha}_j^\top \boldsymbol{\theta}_i + d_2)\right]}, \\
&\vdots \\
\Phi(x_{ij} \geq C_j|\boldsymbol{\theta}_i, \boldsymbol{\alpha}_j, \mathbf{d}_j) &= 0
\end{aligned}$$

These boundaries lead to the conditional probability for the response $x_{ij} = k$ to be

$$\Phi(x_{ij} = k|\boldsymbol{\theta}_i, \boldsymbol{\alpha}_j, \mathbf{d}_j) = \Phi(x_{ij} \geq k|\boldsymbol{\theta}_i, \boldsymbol{\alpha}_j, \mathbf{d}_j) - \Phi(x_{ij} \geq k+1|\boldsymbol{\theta}_i, \boldsymbol{\alpha}_j, \mathbf{d}_j) \tag{2}$$

Note that the 3PL is in fact a special case of the ordinal response model (with the inclusion of a lower asymptote parameter) and can be defined with boundaries in the same fashion, where (2) would consist of only two possible values: $[1 - \Phi(x_{ij} = 1|\boldsymbol{\theta}_i, \boldsymbol{\alpha}_j, d_j, \gamma_j)]$ and $\Phi(x_{ij} = 1|\boldsymbol{\theta}_i, \boldsymbol{\alpha}_j, d_j, \gamma_j)$. Recognizing this, and letting $\boldsymbol{\Psi}$ be the collection of all item parameters, allows us to declare the likelihood equations more concisely.

Expressing the data in indicator form, where

$$\chi(x_{ij}) = \begin{cases} 1, & \text{if } x_{ij} = k \\ 0, & \text{otherwise} \end{cases}$$

the conditional distribution for the ith $n \times 1$ response pattern vector, $\mathbf{x}_i$, is

$$L_\ell(\mathbf{x}_i|\boldsymbol{\Psi}, \boldsymbol{\theta}) = \prod_{j=1}^{n} \prod_{k=0}^{C_j-1} \Phi(x_{ij} = k|\boldsymbol{\Psi}, \boldsymbol{\theta}_i)^{\chi(x_{ij})} \tag{3}$$

Assuming a distributional form $g(\boldsymbol{\theta})$ (most often a multivariate normal) the marginal distribution becomes

$$P_\ell(\boldsymbol{\Psi}|\mathbf{x}) = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} L_\ell(\mathbf{x}_i|\boldsymbol{\Psi}, \boldsymbol{\theta}) g(\boldsymbol{\theta}) d\boldsymbol{\theta} \tag{4}$$

where there are m-fold integrals to be evaluated. Finally, this brings us to the observed data likelihood function. Letting $\mathbf{X}$ represent the complete $N \times n$ data matrix, the observed likelihood equation is

$$L(\mathbf{\Psi}|\mathbf{X}) = \prod_{i=1}^{N} \left[\int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} L_\ell(\mathbf{x}_i|\mathbf{\Psi}, \boldsymbol{\theta}) g(\boldsymbol{\theta}) d\boldsymbol{\theta} \right] \tag{5}$$

2.1. Exploratory and confirmatory item analysis

IRT can be applied in a way that is analogous to exploratory and confirmatory factor analysis for continuous variables (McDonald 1999). Historically, IRT models began in a confirmatory spirit by modeling the item response probabilities as a function of a single underlying factor, with varying degrees of how the item slopes (e.g., Rasch versus 2PL) and intercepts (e.g., ordinal, nominal, or partial credit) were related. But IRT can also be applied in an exploratory manner, whereby the number of dimensions are not assumed known beforehand, and are instead estimated empirically by comparing nested models (Bock and Aitkin 1981) or by rotating the factor loadings matrix to find a more parsimonious structure (Bock, Gibbons, and Muraki 1988). The **TESTFACT** program (Wood *et al.* 2003) was specifically designed for this approach, but other software exist that use different methods of estimation, such as **NOHARM** (Fraser 1998) and **Mplus**'s various WLS estimators (Muthén and Muthén 2008), which use limited-information algorithms, and **BMIRT** (Yao 2008) which uses Bayesian MCMC estimation techniques.

Confirmatory item analysis is useful when more than one factor is thought to be present in the data but various constraints (such as zero slopes) should be imposed. One of the first approaches in this spirit was the bifactor method (Holzinger and Swineford 1937) explicated by Gibbons and Hedeker (1992) for dichotomous data. The inspiration for bifactor models is that a single factor is believed to be present in all items, but with additional clusters of local dependencies formed by other independent specific factors. This approach was later generalized to polytomous data (Gibbons *et al.* 2007) and further expanded to accommodate more than one local dependency caused by specific factors (Cai 2010c).

A more general approach that accommodates linear constraints and missing data can be found in stochastic estimation techniques, such as Bayesian MCMC estimation with Metropolis-Hastings sampling (Metropolis, Rosenbluth, Teller, and Teller 1953; Hastings 1970), Gibbs sampling (Casella and George 1992), or by employing the Metropolis-Hastings Robbins-Monro (MH-RM) algorithm (Cai 2010b). The MH-RM algorithm is explained in more detail below since it is implemented in the **mirt** package.

3. Parameter estimation

IRT parameter estimation has been a progressive science over that past 60 years, moving from heuristic estimation techniques to more advanced Bayesian MCMC methods (Baker and Kim 2004). The early focus was on estimating the item specific parameters for unidimensional models, and until Bock and Aitkin (1981) introduced an EM based estimation solution IRT, applications were largely limited to small testing situations (Baker and Kim 2004). The EM algorithm, which was introduced by using fixed Gauss-Hermite quadrature, appeared to be a reasonable solution for lower dimensional models without compromising numerical accuracy. Unfortunately this technique quickly becomes inefficient as the number of dimensions increases, since the number of quadrature points required for estimating the 'E-step' increases exponentially and must be accommodated for by decreasing the number of quadrature. A partial solution for a moderate number of dimensions was described by Schilling and Bock (2005), where the authors demonstrated that adaptive quadrature could be used for better accuracy when a smaller number of quadratures per dimension is used, but the problem of high-dimensional solutions still remained.

More recently, a solution to the high-dimensionality problem has been to employ stochastic estimation methods for exploratory and confirmatory item analysis. Bayesian MCMC methods have been explored by Edwards (2010) and Sheng (2010), and both authors have released software to estimate the parameters for polytomous and dichotomous response models, respectively. These methods are not implemented in the **mirt** package, so they will not be discussed further, but see Bolt (2005) and Wirth and Edwards (2007) for more thorough reviews of using these full-information estimation methods and for item response model estimation in general. The two methods that are implemented in the **mirt** package are the fixed quadrature EM method for exploratory (Bock *et al.* 1988; Muraki and Carlson 1995) and bifactor (Gibbons and Hedeker 1992; Gibbons *et al.* 2007) models, and the Metropolis-Hastings Robbins-Monro method for exploratory (Cai 2010a) and confirmatory (Cai 2010b) polytomous models.

3.1. Estimation using the expectation-maximization algorithm

Bock and Aitkin (1981) were the first to propose a feasible method for estimating the item parameters of large scale tests using a method similar to the Expectation-Maximization (EM) algorithm (Dempster, Laird, and Rubin 1977). As explained in Bock *et al.* (1988) and Muraki and Carlson (1995) this method is appropriate for low to moderate factor solutions, so long as the number of quadratures per dimension decreases as the number of factors increases. For the following EM estimation methods we will explore only the special case when all the data are dichotomous. To begin, approximate (4) for each unique response vector by using m-fold Gauss-Hermite quadrature

$$\tilde{P}_\ell = \sum_{qm=1}^{Q} \cdots \sum_{q2}^{Q} \sum_{q1}^{Q} L_\ell(\mathbf{x}_\ell | \mathbf{\Psi}, \mathbf{K}) g(K_{q1}) g(K_{q2}) \ldots g(K_{qm}) \tag{6}$$

From this result the observed likelihood based on the u unique response patterns with r_u individuals with identical patterns becomes

$$L(\mathbf{\Psi}|\mathbf{X}) = \frac{N!}{r_1!\, r_2! \, \ldots \, r_u!} \tilde{P}_1^{r1}\, \tilde{P}_2^{r2} \, \ldots \, \tilde{P}_u^{ru} \tag{7}$$

Differentiating with respect to an arbitrary item parameter within item j and integrating out the m-dimensions of $\boldsymbol{\theta}$ gives

$$\frac{\partial \log L(\mathbf{\Psi}|\mathbf{X})}{\partial \psi_j} = \int_{-\infty}^{\infty} \cdots \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\bar{r}_j - \bar{N}\Phi_j(\boldsymbol{\theta})}{\Phi_j(\boldsymbol{\theta})[1-\Phi_j(\boldsymbol{\theta})]} \cdot \frac{\partial \Phi_j(\boldsymbol{\theta})}{\partial \psi_j} g(\boldsymbol{\theta}) d(\boldsymbol{\theta}) \tag{8}$$

where

$$\bar{r}_j = \sum_{\ell=1}^{u} \frac{r_\ell x_{\ell j} L_\ell(\mathbf{x}_i|\mathbf{\Psi}, \boldsymbol{\theta})}{\tilde{P}_\ell}, \tag{9}$$

and

$$\bar{N} = \sum_{\ell=1}^{u} \frac{r_\ell L_\ell(\mathbf{x}_i|\mathbf{\Psi}, \boldsymbol{\theta})}{\tilde{P}_\ell}. \tag{10}$$

Approximating (8) by using quadratures gives

$$\frac{\partial \log L(\mathbf{\Psi}|\mathbf{X})}{\partial \psi_j} = \sum_{qm}^{Q} \cdots \sum_{q2}^{Q} \sum_{q1}^{Q} \frac{\bar{r}_j - \bar{N}\Phi_j(\mathbf{X})}{\Phi_j(\mathbf{X})[1-\Phi_j(\mathbf{X})]} \cdot \frac{\partial \Phi_j(\mathbf{X})}{\partial \psi_j} g(K_{q1})g(K_{q2})\ldots g(K_{qm}) \tag{11}$$

The 'E-step' of the EM algorithm consists of finding (9) and (10) by treating $\mathbf{\Psi}$ as provisionally known when computing $L_\ell(\mathbf{K})$. The 'M-step' then consists of finding the $\mathbf{0}$ root of (11) independently for each item. The EM process is repeated until the change between iterations falls below some pre-specified tolerance.

A special case for EM estimation: The bifactor model

The full-information bifactor model (Gibbons and Hedeker 1992; Gibbons *et al.* 2007) combines a unidimensional model with the so-called 'simple structure' item loadings model (Thurstone 1947). The purpose is to estimate a common latent trait alongside independent components for each item so that local dependencies are accounted for properly. The bifactor model can specify only one additional item specific factor (although see Cai 2010c), but is not limited in the number of factors estimated since the quadratures remain fixed regardless of the number of specific factors extracted.

Define P_ℓ as

$$P_\ell = \int_{-\infty}^{\infty} \left(\prod_{s=2}^{m} \int_{-\infty}^{\infty} L_\ell(\theta_1, \theta_s) g(\theta_s) d\theta_s \right) g(\theta_1) d\theta_1 \tag{12}$$

where $L_\ell(\theta_1, \theta_s) = \prod_{j=1}^{n} ([\Phi_{js}(\theta_1, \theta_s)]^{x_{\ell j}} [1 - \Phi_{js}(\theta_1, \theta_s)]^{1-x_{\ell j}})^{c_{js}}$, and c_{jm} indexes the nonzero loading of item j on dimension m, where only one value in c_{jm} is equal to 1, otherwise $c_{jm} = 0$. The gradient with respect to an arbitrary parameter from item j, expressed in quadrature form, is then

$$\frac{\partial \log L(\mathbf{\Psi}|\mathbf{X})}{\partial \psi_j} \cong \sum_{q1}^{Q} \sum_{s=2}^{m} c_{js} \sum_{qs}^{Q} \frac{\bar{r}_{js}(\mathbf{K}) - \bar{N}_s(\mathbf{K})\Phi_{js}(\mathbf{K})}{\Phi_{js}(\mathbf{K})[1-\Phi_{js}(\mathbf{K})]} \cdot \frac{\partial \Phi_{js}(\mathbf{K})}{\partial \psi_j} g(K_{qs})g(K_{q1}) \tag{13}$$

where

$$E_{\ell m}(\theta_1) = \frac{\prod_{s=2}^{m} \int_{\theta_s} L_{\ell s}(\theta_1, \theta_s) g(\theta_s) d\theta_s}{\int_{\theta_s} L_{\ell m}(\theta_1, \theta_m) g(\theta_m) d\theta_m}, \tag{14}$$

$$\bar{r}_{jm}(\mathbf{K}) = \sum_{\ell=1}^{u} r_\ell x_{\ell j} [E_{\ell m}(K_{q1})] \frac{L_\ell(K_{q1}, K_{qm})}{\tilde{P}_\ell}, \tag{15}$$

and

$$\bar{N}_m(\mathbf{K}) = \sum_{\ell=1}^{u} r_\ell [E_{\ell m}(K_{q1})] \frac{L_\ell(K_{q1}, K_{qm})}{\tilde{P}_\ell}. \tag{16}$$

As before, (15) and (16) are computed by treating $\mathbf{\Psi}$ as provisionally known, and (13) is then solved for each item independently. This model has the benefit of having a fixed number of quadratures regardless of the number of specific factors estimated, and is closely related to 'Testlet' response models (Wainer, Bradlow, and Wang 2007).

3.2. Estimation via the Metropolis-Hastings Robbins-Monro algorithm

The Metropolis-Hastings Robbins-Monro (MH-RM) algorithm estimates the item parameters by using a stochastically imputed complete-data likelihood with an assumed population distributional form (typically, multivariate normal)

$$L(\mathbf{\Psi}|\mathbf{X}, \boldsymbol{\theta}) = \prod_{i=1}^{N} L_\ell(\mathbf{x}_i|\mathbf{\Psi}, \boldsymbol{\theta}) g(\boldsymbol{\theta}|\mu, \mathbf{\Sigma}). \tag{17}$$

For exploratory item factor analysis the population mean vector μ is usually assumed to be a fixed $m \times 1$ vector of zeros, and $\mathbf{\Sigma}$ is assumed to be a fixed $m \times m$ identity matrix. However, in confirmatory item analysis various elements in μ and $\mathbf{\Sigma}$ may be estimated in a way analogous to confirmatory factor analysis in the structural equation modeling framework (e.g., see Bollen 1989, Chapter 7). As can be seen in (17) the complete-data log-likelihood is composed of two additive components: the log-likelihood for a multivariate ordinal regression and the log-likelihood relationship between the factor scores.

The MH-RM algorithm deals with the integration problem in a different way than the traditional EM approach. For the EM algorithm $\boldsymbol{\theta}$ is treated as set of 'nuance' parameters with a known distribution and are then integrated out of the likelihood equation (using numerical quadrature) so that the marginal frequencies ($\bar{r}_j$ and $\bar{N}_j$) can be approximated. The marginal frequencies are then used to update the item level parameters, and with the newly updated parameters the process is repeated. The MH-RM and related methods (such as the stochastic approximated EM) use stochastic methods to temporarily fill in the missing $\boldsymbol{\theta}$ instead, and once filled in the values are treated as if they were 'known'. Given the newly augmented data the item parameters can then be updated by using conventional root-finding methods that use the complete-data log-likelihood function directly. Imputation methods are not exact or determinate but often allow for easier and more convenient evaluation of higher dimensional integrals than their numerical quadrature counterparts. The MH-RM is a more recent attempt to control for the inaccuracies borne out of using stochastic imputations to approximate the $\boldsymbol{\theta}$ parameters.

Cai (2010a) demonstrated that when using a stochastically imputed complete-data model, and properly accounting for the error in the stochastic imputations, maximum-likelihood estimation and observed parameter standard errors can be calculated. The estimation begins by computing $\boldsymbol{\theta}^{(d)}$, given initial start values in $\mathbf{\Psi}$, by using a Metropolis-Hastings sampler (Metropolis *et al.* 1953; Hastings 1970). This allows the complete-data gradient vector and hessian matrix to be calculated and utilized in updating the initial parameters[2]. The initial parameters are then updated for the next iteration by using a single Newton-Raphson correction and these new parameters are then used to sample $\boldsymbol{\theta}^{(d+1)}$. This process is repeated for several cycles and constitutes the so-called 'burn-in' period. After sufficient burn-in iterations, a set of parameter estimates are reserved to compute the approximate starting values

[2]Multiple sets of $\boldsymbol{\theta}$'s can be drawn and averaged over for constructing the gradient and hessian, but for item factor analysis this is rarely required.

for the MH-RM algorithm. Finally, the last set of parameter updates are conducted using the same method as before, but are now controlled by using the Robbins-Monro (Robbins and Monro 1951) root finding algorithm, which slowly converges as the decreasing gain constant approaches zero. In this way, the inaccuracies borne from the Metropolis-Hastings sampler are properly accounted for when attempting to maximize the parameter estimates. The MH-RM algorithm is useful for estimating both exploratory and confirmatory item response models. For specifying confirmatory models, several linear restrictions can be imposed on single parameters (e.g., $\alpha_{1j} = 0$) or between parameters (e.g., $\alpha_{1j} \equiv \alpha_{2j}$). This is accomplished by defining a matrix $\mathbf{L}$, which selects the parameters that are to be estimated, and a vector $\mathbf{c}$, which contains the fixed values for the parameters. For example,

$$\boldsymbol{\Psi}_r = \mathbf{c} + \mathbf{L}(\boldsymbol{\Psi}^{(v)}) \tag{18}$$

$$= \begin{pmatrix} 0 \\ 0 \\ 0.1 \\ \vdots \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ d_1 \\ \gamma_1 \\ \vdots \\ \mu \\ \sigma \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ d_1 \\ 0.1 \\ \vdots \\ 0 \\ 1 \end{pmatrix}$$

where $\boldsymbol{\Psi}^{(v)}$ is a *vech* stacked vector containing all of the possible group and item parameters. This example shows that the slope and intercept are estimated for item 1, the guessing parameter is fixed at 0.1, and the latent mean and variance are fixed at 0 and 1, respectively. The usefulness of this formulation lies in how to manipulate the gradient and hessian given the user defined restrictions, since the restricted gradient

$$\frac{\partial \log L(\boldsymbol{\Psi}|\mathbf{X},\boldsymbol{\theta})}{\partial \boldsymbol{\Psi}_r^\top} = \frac{\partial \log L(\boldsymbol{\Psi}|\mathbf{X},\boldsymbol{\theta})}{\partial \boldsymbol{\Psi}^\top}\mathbf{L}, \tag{19}$$

and hessian

$$\frac{\partial^2 \log L(\boldsymbol{\Psi}|\mathbf{X},\boldsymbol{\theta})}{\partial \boldsymbol{\Psi}_r \partial \boldsymbol{\Psi}_r^\top} = \mathbf{L}^\top \frac{\partial^2 \log L(\boldsymbol{\Psi}|\mathbf{X},\boldsymbol{\theta})}{\partial \boldsymbol{\Psi} \partial \boldsymbol{\Psi}^\top}\mathbf{L} \tag{20}$$

can be easily expressed. This result implies that one can first estimate the complete-data gradient and hessian given no restrictions, and simply apply (19) and (20) to update only a subset of the parameters that are to be estimated.

The MH-RM algorithm offers a flexible way to specify both confirmatory and exploratory item models but has three main shortcomings: the estimation times will be much larger for lower dimensional problems when compared to an EM solution, the observed data log-likelihood is not calculated automatically and must be estimated use Monte Carlo integration, and parameter estimates will not be identical between different estimations. It is recommended to use the MH-RM algorithm only when the dimensionality increases to the point where the EM solution becomes difficult due to the large number of quadratures required (approximately around four or more factors), or when a multidimensional confirmatory model is required. However, the user should keep in mind that even the MH-RM method will become slower as the dimensions increase, so testing parameters in, for example, a bifactor model with eight specific factors may take an extended amount of time.

4. Implementation in the mirt package

Four separate datasets are used to demonstrate how each major estimation function typically behaves. These are: the well known 5-item LSAT7 data (Bock and Lieberman 1970); the SAT12 data set, which is available as an example from the **TESTFACT** (Wood *et al.* 2003) manual; a simulated data set constructed from the item parameters with orthogonal factors found in Reckase (2009, p. 153); and a simulated set derived from the parameters shown in Appendix B. All simulated data were constructed by using `simdata()` from the **mirt** package.

There are four main functions for estimating MIRT models: `mirt()`, `bfactor()`, `polymirt()`, and `confmirt()`, where the latter two employ the MH-RM algorithm. Individual item plots can be generated using `itemplot()`, factor scores (with MAP or EAP estimation) can be estimated using the `fscores()` function, and MIRT data with known parameters can be simulated using `simdata()`. The subroutines that do not directly relate to model estimation are demonstrated in the appendices.

The data that are passed to all the estimation functions must be either a `matrix` or `data.frame` consisting of numeric values for the item responses. For example, coding an 'ability' test as 0 for incorrect and 1 for correct, or coding a Likert type format with 1 representing the lowest category and 5 as the highest category are conceptually the preferred layout, although technically the choice of direction is arbitrary. Responses that are omitted must be coded as `NA`. Only complete data-sets can be passed to these functions, so if the data are in a tabulated format (see below) the use of `expand.table()` can quickly create the appropriate input.

4.1. An example with the `mirt()` function

The LSAT7 data found in Bock and Lieberman (1970) initially were presented in tabulated form, with the number of individuals with identical response pattern in the rightmost column. This type of input can be modified easily with the `expand.table()`, and here we see the default use of `mirt()` for a one-factor model

```
R> library("mirt")
R> data <- expand.table(LSAT7)
R> (mod1 <- mirt(data, 1))

Call:
mirt(data = data, nfact = 1)

Full-information factor analysis with 1 factor
Converged in 18 iterations using 40 quadrature points.
Log-likelihood = -2657.038
AIC = 5334.076
BIC = 5383.153
G^2 = 31.71, df = 21, p = 0.0626, RMSEA = 0.023
```

which converges quickly. Using `coef()` extracts the maximum-likelihood parameters and item facilities (i.e., average number of correct responses), while `summary()` transforms the slopes into the traditional factor analysis loadings metric (see Bock *et al.* 1988).

```
R> coef(mod1)
```

```
Parameter slopes and intercepts:

         a_1   d_1 guess
Item.1 0.584 1.093     0
Item.2 0.634 0.475     0
Item.3 0.993 1.054     0
Item.4 0.452 0.286     0
Item.5 0.436 1.091     0
```

The values is the first column (`a_1`) reflect the item slopes for factor one, while the values is the second column (`d_1`) correspond to the item intercept. The names of these columns also reflects their relationship to equation (1), although the γ parameter has been renamed to `guess` instead to avoid confusion.

```
R> summary(mod1)

Unrotated factor loadings:

         F_1    h2
Item.1 0.504 0.254
Item.2 0.536 0.287
Item.3 0.705 0.496
Item.4 0.412 0.170
Item.5 0.400 0.160

SS loadings:  1.367
Proportion Var:  0.273
```

The object returned by `mirt()` has various diagnostic tools available to determine where the model may be malfunctioning. By default, `residuals()` computes the local dependence (LD) pairwise statistic between each pair of items, which is very similar to a signed χ^2 value (Chen and Thissen 1997). Also, a standardized version of the LD statistic (Cramer's V) is printed above the diagonal to aid in interpretation when items contain more than two response options and hence more degrees of freedom (i.e., objects return by `polymirt()` and `confmirt()`). The residuals can also be in the form of the marginal expected frequencies for each response pattern by specifying the input the option `restype = "exp"`.

```
R> residuals(mod1)

LD matrix:

       Item.1 Item.2 Item.3 Item.4 Item.5
Item.1     NA  0.022  0.029  0.050  0.048
Item.2 -0.479     NA  0.034  0.017  0.038
Item.3 -0.855  1.149     NA  0.012  0.002
Item.4  2.474 -0.282 -0.149     NA  0.001
Item.5  2.286 -1.433 -0.004 -0.001     NA
```

```
R> residuals(mod1, restype = "exp")

      Item.1 Item.2 Item.3 Item.4 Item.5 Freq     exp    res
 [1,]      0      0      0      0      0   12  10.142  0.584
 [2,]      0      0      0      0      1   19  18.462  0.125
 [3,]      0      0      0      1      0    1   4.500 -1.650
 [4,]      0      0      0      1      1    7  10.608 -1.108
 [5,]      0      0      1      0      0    3   4.992 -0.891
 [6,]      0      0      1      0      1   19  15.942  0.766
 [7,]      0      0      1      1      0    3   3.966 -0.485
 [8,]      0      0      1      1      1   17  16.414  0.145
 [9,]      0      1      0      0      0   10   3.964  3.031
[10,]      0      1      0      0      1    5  10.355 -1.664
[11,]      0      1      0      1      0    3   2.557  0.277
[12,]      0      1      0      1      1    7   8.609 -0.548
[13,]      0      1      1      0      0    7   4.418  1.228
[14,]      0      1      1      0      1   23  20.381  0.580
[15,]      0      1      1      1      0    8   5.142  1.261
[16,]      0      1      1      1      1   28  31.516 -0.626
[17,]      1      0      0      0      0    7  12.773 -1.615
[18,]      1      0      0      0      1   39  32.440  1.152
[19,]      1      0      0      1      0   11   8.002  1.060
[20,]      1      0      0      1      1   34  26.188  1.527
[21,]      1      0      1      0      0   14  13.346  0.179
[22,]      1      0      1      0      1   51  59.739 -1.131
[23,]      1      0      1      1      0   15  15.053 -0.014
[24,]      1      0      1      1      1   90  89.284  0.076
[25,]      1      1      0      0      0    6   8.088 -0.734
[26,]      1      1      0      0      1   25  29.363 -0.805
[27,]      1      1      0      1      0    7   7.340 -0.126
[28,]      1      1      0      1      1   35  34.756  0.041
[29,]      1      1      1      0      0   18  19.431 -0.325
[30,]      1      1      1      0      1  136 130.132  0.514
[31,]      1      1      1      1      0   32  33.306 -0.226
[32,]      1      1      1      1      1  308 308.790 -0.045
```

Estimating more than one factor with `mirt()` is performed simply by changing the second numeric input value. There are several areas that should be considered when increasing the number of dimensions to extract. To begin, by default the number of quadrature values used during estimation decreases so that estimation time is lower, and so that there are not any memory leaking problems. However, while this means that solutions using `mirt()` do not take as long to estimate, it does mean that the accuracy of estimating the parameters will suffer. For moderate to high-dimensional solutions it may be better to use the `polymirt()` and `confmirt()` functions (see below).

```
R> (mod2 <- mirt(data, 2))
```

```
Call:
mirt(data = data, nfact = 2)

Full-information factor analysis with 2 factors
Converged in 11 iterations using 15 quadrature points.
Log-likelihood = -2652.096
AIC = 5334.192
BIC = 5407.808
G^2 = 21.82, df = 17, p = 0.1915, RMSEA = 0.017
```

Again, the coefficients can be extracted as above, but now `summary()` holds a different purpose. Since the orientation of the factor loadings is arbitrary the initially extracted solution should be rotated to a simpler structure to better facilitate interpretation. The default rotation method is the varimax criterion, but many other rotations available in the **GPArotation** package (Bernaards and Jennrich 2005) are integrated into the function to save the user time and effort. For example, an oblimin rotated factor solution, suppressing absolute loadings less than 0.25, is

```
R> summary(mod2, rotate = "oblimin", suppress = 0.25)
```

```
Rotation:  oblimin

Rotated factor loadings:

         F_1    F_2    h2
Item.1    NA -0.711 0.493
Item.2 0.545     NA 0.305
Item.3 0.759     NA 0.572
Item.4    NA -0.292 0.180
Item.5    NA -0.307 0.173

Factor correlations:

       F_1    F_2
F_1  1.000 -0.591
F_2 -0.591  1.000

Rotated SS loadings:  0.929 0.686
```

Nested model comparison can be performed using the `anova()` generic, returning a likelihood-ratio χ^2 test as well as returning the difference in AIC and BIC values. As seen below, the difference between `mod1` and `mod2` is marginally-significant at the $\alpha = 0.05$ cut-off, while the AIC and BIC decrease indicate that the extra parameters estimated likely do not contribute additional useful information.

```
R> anova(mod1, mod2)
```

```
Chi-squared difference:

X2 = 9.884, df = 4, p = 0.0424
AIC difference =  -0.116
BIC difference =  -24.655
```

Finally, `mirt()` contains plotting features used for didactic illustration (see Appendix A), but in general are not as flexible as the **plink** package (Weeks 2010). The generic functions demonstrated in this section are applicable for the remaining estimation methods as well, and essentially perform the same behavior. For more detailed information refer to the documentation found in the **mirt** package.

4.2. An example with the `bfactor()` function

Next we examine a bifactor model for the SAT12 data. First, we must change the raw data in SAT12 into dichotomous (correct-incorrect) form by using the `key2binary()` function. From here we declare which specific-factor affects which item in numeric sequence vector, where each element corresponds to its respective item. Here, for example, item one is a function of the general factor and of specific factor 2, while item two is influenced by specific factor 3 and the general factor, etcetera. As an added feature not mentioned before, fixed values for the lower asymptotes may be specified a priori for all estimation functions.

If number of factors is greater than one then multivariate discrimination and intercepts are also available. The multivariate discrimination and intercepts are defined as

$$MVDISC = \left(\sum_{s=1}^{m} \alpha_{ns}^2 \right)^{1/2} \tag{21}$$

and

$$MVINT_k = \frac{-d_k}{MVDISC}, \tag{22}$$

respectively. These indices are potentially useful for determining an item's overall utility across factors (Reckase and McKinley 1991), and are printed for all **mirt** objects when using `coef()`.

```
R> data("SAT12")
R> data <- key2binary(SAT12, key = c(1, 4, 5, 2, 3, 1, 2, 1, 3, 1, 2, 4, 2,
+    1, 5, 3, 4, 4, 1, 4, 3, 3, 4, 1, 3, 5, 1, 3, 1, 5, 4, 5))
R> specific <- c(2, 3, 2, 3, 3, 2, 1, 2, 1, 1, 1, 3, 1, 3, 1, 2, 1, 1, 3,
+    3, 1, 1, 3, 1, 3, 3, 1, 3, 2, 3, 1, 2)
R> guess <- rep(0.1, 32)
R> b_mod1 <- bfactor(data, specific, guess)
R> coef(b_mod1)

Parameters with multivariate discrimination and intercept:
```

```
          a_G    a_1    a_2    a_3    d_1 guess mvdisc mvint_1
Item.1  0.699     NA  0.403     NA -1.081   0.1  0.807   1.340
Item.2  1.003     NA     NA  0.546  0.087   0.1  1.142  -0.076
Item.3  1.367     NA -0.243     NA -1.511   0.1  1.388   1.088
Item.4  0.433     NA     NA  0.394 -0.586   0.1  0.585   1.000
Item.5  0.613     NA     NA  0.328  0.239   0.1  0.695  -0.344
Item.6  1.685     NA  0.327     NA -2.838   0.1  1.716   1.654
Item.7  0.619  0.634     NA     NA  0.828   0.1  0.886  -0.934
Item.8  1.113     NA  0.898     NA -2.246   0.1  1.430   1.570
Item.9  0.239  0.582     NA     NA  1.346   0.1  0.629  -2.141
Item.10 0.789  0.594     NA     NA -0.520   0.1  0.988   0.527
Item.11 0.914  0.512     NA     NA  3.142   0.1  1.047  -3.000
Item.12 0.070     NA     NA  0.199 -0.380   0.1  0.211   1.803
Item.13 0.668  0.384     NA     NA  0.398   0.1  0.771  -0.516
Item.14 0.673     NA     NA  0.603  0.680   0.1  0.904  -0.752
Item.15 0.813  0.394     NA     NA  1.112   0.1  0.903  -1.232
Item.16 0.554     NA  0.356     NA -0.470   0.1  0.659   0.713
Item.17 0.881  0.212     NA     NA  2.412   0.1  0.906  -2.662
Item.18 1.559  0.165     NA     NA -1.069   0.1  1.568   0.682
Item.19 0.542     NA     NA  0.028 -0.004   0.1  0.543   0.008
Item.20 0.861     NA     NA  0.270  1.450   0.1  0.903  -1.606
Item.21 0.306  0.437     NA     NA  1.514   0.1  0.534  -2.838
Item.22 0.888  0.049     NA     NA  1.985   0.1  0.890  -2.231
Item.23 0.533     NA     NA  0.406 -0.871   0.1  0.670   1.301
Item.24 0.767  0.161     NA     NA  0.650   0.1  0.784  -0.830
Item.25 0.607     NA     NA  0.582 -0.677   0.1  0.841   0.805
Item.26 1.170     NA     NA  0.379 -0.404   0.1  1.229   0.329
Item.27 1.079  0.286     NA     NA  1.534   0.1  1.116  -1.374
Item.28 0.713     NA     NA  0.046 -0.064   0.1  0.715   0.090
Item.29 0.981     NA  1.100     NA -1.145   0.1  1.474   0.777
Item.30 0.274     NA     NA -0.108 -0.316   0.1  0.294   1.074
Item.31 1.722 -0.213     NA     NA  1.797   0.1  1.735  -1.036
Item.32 0.131     NA  0.016     NA -1.578   0.1  0.132  11.966
```

In this example all of the item parameters appear to have converged to reasonable values, however this will not always be the case. When intercept parameters appear to be excessively large we have a few options to help alleviate the problem: remove the guessing values by fixing them back to 0, or by placing prior distribution constraints on the intercepts to try to keep the values from increasing too much. Below is the latter option, where a weak normal prior ($N \sim (0, 4)$) is imposed on the intercept for `Item.32` so that the item parameters are estimated with a Bayesian MAP method instead of by maximum-likelihood (see Bock *et al.* 1988, for more details). Although not demonstrated below it is also possible to impose β priors on the slope parameters if there are slope related convergence problems.

```
R> b_mod2 <- bfactor(data, specific, guess, par.prior = list(int = c(0, 4),
+    int.items = 32))
```

```
Intercept prior for item(s): 32
```

```
R> coef(b_mod2)

Parameters with multivariate discrimination and intercept:

          a_G    a_1    a_2    a_3    d_1 guess mvdisc mvint_1
Item.1  0.703     NA  0.387     NA -1.082   0.1  0.803   1.348
Item.2  1.006     NA     NA  0.550  0.091   0.1  1.147  -0.079
Item.3  1.353     NA -0.246     NA -1.502   0.1  1.376   1.092
Item.4  0.433     NA     NA  0.385 -0.585   0.1  0.580   1.009
Item.5  0.618     NA     NA  0.332  0.241   0.1  0.702  -0.343
Item.6  1.749     NA  0.316     NA -2.914   0.1  1.777   1.639
Item.7  0.614  0.641     NA     NA  0.833   0.1  0.887  -0.938
Item.8  1.124     NA  0.885     NA -2.242   0.1  1.431   1.567
Item.9  0.244  0.579     NA     NA  1.345   0.1  0.628  -2.142
Item.10 0.785  0.590     NA     NA -0.516   0.1  0.982   0.525
Item.11 0.923  0.546     NA     NA  3.186   0.1  1.073  -2.971
Item.12 0.069     NA     NA  0.199 -0.379   0.1  0.211   1.800
Item.13 0.665  0.386     NA     NA  0.401   0.1  0.769  -0.521
Item.14 0.670     NA     NA  0.603  0.675   0.1  0.901  -0.749
Item.15 0.805  0.409     NA     NA  1.120   0.1  0.903  -1.239
Item.16 0.558     NA  0.355     NA -0.469   0.1  0.662   0.709
Item.17 0.876  0.207     NA     NA  2.416   0.1  0.900  -2.684
Item.18 1.542  0.165     NA     NA -1.059   0.1  1.551   0.683
Item.19 0.538     NA     NA  0.029 -0.008   0.1  0.539   0.015
Item.20 0.858     NA     NA  0.298  1.458   0.1  0.908  -1.606
Item.21 0.303  0.423     NA     NA  1.509   0.1  0.521  -2.898
Item.22 0.885  0.067     NA     NA  1.988   0.1  0.888  -2.239
Item.23 0.535     NA     NA  0.404 -0.873   0.1  0.671   1.301
Item.24 0.763  0.170     NA     NA  0.650   0.1  0.781  -0.832
Item.25 0.613     NA     NA  0.586 -0.683   0.1  0.848   0.805
Item.26 1.174     NA     NA  0.363 -0.400   0.1  1.229   0.326
Item.27 1.069  0.287     NA     NA  1.530   0.1  1.107  -1.382
Item.28 0.720     NA     NA  0.036 -0.062   0.1  0.721   0.086
Item.29 1.003     NA  1.132     NA -1.167   0.1  1.512   0.771
Item.30 0.275     NA     NA -0.112 -0.314   0.1  0.297   1.060
Item.31 1.738 -0.212     NA     NA  1.816   0.1  1.751  -1.037
Item.32 0.139     NA  0.004     NA -1.575   0.1  0.139  11.351
```

As we can see the intercept parameter appears to be pulled slightly towards 0 which helps to add numerical stability to situations where the intercepts appear to be approaching $-\infty$ or ∞.

4.3. An example with the `polymirt()` function

`polymirt()` and `confmirt()` both estimate the model parameters with the MH-RM algorithm. `polymirt()` is applicable for exploratory item analysis for dichotomous and polytomous data, and the object returned has many commonalities with objects returned by `mirt()`. There are a few pros and cons to using these stochastic functions, the pros being that parameter standard errors are automatically computed as a by-product of estimation, the models stay accurate even with higher dimensionality, lower asymptotes may be estimated (with β priors automatically added), and in `confmirt()` various item constraints can be imposed. The cons are that the time to estimate these models will be longer than `mirt()` or `bfactor()` for low-dimensional models since the actual estimation of the parameters takes more time, computation of such useful statistics as the observed log-likelihood must be estimated by Monte Carlo methods, and the parameters will vary slightly between independent estimations. However the added estimation time is not a major concern since the overall execution times often fall well within reasonable limits (see below).

Specification of a three-dimensional exploratory factor analysis model for the first simulated data-set is

```
R> p_mod <- polymirt(simdata1, 3)

Stage 1: Cycle = 10, Log-Lik = -37423.2, Max Change = 0.0626
Stage 1: Cycle = 20, Log-Lik = -36982.9, Max Change = 0.0343
Stage 1: Cycle = 30, Log-Lik = -36881.0, Max Change = 0.0446
........
Stage 3: Cycle = 350, Log-Lik = -36912.7, gam = 0.008, Max Change = 0.0009
Stage 3: Cycle = 360, Log-Lik = -37018.3, gam = 0.008, Max Change = 0.0012

Calculating log-likelihood...

R> p_mod

Call:
polymirt(data = simdata1, nfact = 3)

Full-information factor analysis with 3 factors
Converged in 363 iterations.
Log-likelihood = -29503.28, SE = 0.064
AIC = 59240.56
BIC = 59895.86
G^2 = 36238.01, df = 1877, p = 0, RMSEA = 0.096

R> coef(p_mod)

Unrotated parameters, multivariate discrimination and intercept:

          a_1    a_2    a_3    d_1 guess mvdisc mvint_1
Item_1  0.513 -0.213  0.486  0.248     0  0.738  -0.336
Item_2  0.370 -0.117  0.323 -0.257     0  0.505   0.509
Item_3  0.868 -0.366  0.498 -0.487     0  1.066   0.457
........
Item_29 0.391  0.311  0.179  0.058     0  0.531  -0.108
Item_30 0.761  0.903 -0.115  0.071     0  1.186  -0.060
```

```
Std. Errors:

           a_1    a_2    a_3    d_1 guess
Item_1  0.0596 0.0530 0.0588 0.0526    NA
Item_2  0.0533 0.0494 0.0522 0.0494    NA
Item_3  0.0778 0.0616 0.0648 0.0614    NA
........
Item_29 0.0532 0.0514 0.0497 0.0485    NA
Item_30 0.0739 0.0794 0.0578 0.0577    NA
```

The behavior of `summary()`, `anova()`, `residuals()`, and `plot()` function the same as before, with this addition of `logLik()` that computes a Monte Carlo estimated integral for the observed log-likelihood. By default, the log-likelihood is approximated with 3000 draws at the end of both estimation methods, but can be suppressed by specifying `calcLL = FALSE` in the function calls.

4.4. An example with the `confmirt()` function

For this example we assume that the form of the loadings, and the relationships among the factors, are known or suspected a priori. Here we will try to recover the parameters used to simulate the data (which can be found in Appendix B). To begin, we must declare where the factors load, the relationships among the loadings, the relationships among the factors, as well as any additional parameter constraints. A model is specified by indicating which latent factors affect which numerically labeled item and by utilizing a select few keywords (e.g., `COV`, `MEAN`, `INT`, `SLOPE`, etc.) for additional parameter relations. For example, the following code declares a two-factor confirmatory model where the first factor (`F1`) affects items 1 to 4 while the second factor (`F2`) affects items 4 to 8 and the `COV` option allows the covariance between `F1` and `F2` to be freely estimated.

```
R> model.1 <- confmirt.model()
  F1 = 1-4
  F2 = 4-8
  COV = F1*F2

Read 12 records

R> c_mod <- confmirt(simdata2, model.1, printcycles = FALSE)

Calculating log-likelihood...

R> c_mod

Call:
confmirt(data = simdata2, model = model.1, printcycles = FALSE)

Full-information item factor analysis with 2 factors
Log-likelihood = -10067.6, SE = 0.038
```

```
AIC = 20179.19
BIC = 20302.41
G^2 = 761.02, df = 319, p = 0, RMSEA = 0.026
Converged in 210 iterations.

R> coef(c_mod)

ITEM PARAMETERS:
        a_F1  a_F2    d_1    d_2    d_3 guess
Item_1 1.446    NA -0.849     NA     NA     0
Item_2 0.701    NA -1.631     NA     NA     0
Item_3 1.027    NA  1.499     NA     NA     0
Item_4 1.076 0.520  0.138     NA     NA     0
Item_5    NA 1.433  2.993 1.9721 -0.409    NA
Item_6    NA 0.550  2.620 1.0829 -0.960    NA
Item_7    NA 1.036  2.026 0.0046     NA    NA
Item_8    NA 0.962  1.061     NA     NA     0

Std. Errors:
         a_F1   a_F2    d_1    d_2    d_3 guess
Item_1 0.1092     NA 0.0787     NA     NA    NA
Item_2 0.0974     NA 0.1123     NA     NA    NA
Item_3 0.1021     NA 0.1060     NA     NA    NA
Item_4 0.0967 0.0744 0.0610     NA     NA    NA
Item_5     NA 0.0847 0.1618 0.1119 0.0636    NA
Item_6     NA 0.0519 0.1780 0.0668 0.0632    NA
Item_7     NA 0.0711 0.1109 0.0553     NA    NA
Item_8     NA 0.0896 0.0804     NA     NA    NA

GROUP PARAMETERS:
Covariance:
      F1    F2
F1 1.000 0.417
F2 0.417 1.000

Std. Errors:
       F1     F2
F1     NA 0.0171
F2 0.0171     NA
```

In this example the MH-RM estimation appears to have recovered the parameters well. Additional options may be used to test a more restricted model by setting various parameters equal or by fixing parameters to constant values. Next, we fix the first item slope to 1.5 with the `SLOPE` command, and set slopes 3-4 on `F1` and 7-8 on `F2` to be equal during estimation.

```
R> model.2 <- confmirt.model()
  F1 = 1-4
  F2 = 4-8
  COV = F1*F2
  SLOPE = F1@1 eq 1.5, F1@3 eq F1@4, F2@7 eq F2@8

Read 12 records

R> c_mod2 <- confmirt(simdata2, model.2, printcycles = FALSE)

Calculating log-likelihood...

R> coef(c_mod2)

ITEM PARAMETERS:
       a_F1  a_F2    d_1     d_2    d_3 guess
Item_1 1.50    NA -0.911      NA     NA     0
Item_2 0.70    NA -1.631      NA     NA     0
Item_3 1.02    NA  1.494      NA     NA     0
Item_4 1.02 0.537  0.134      NA     NA     0
Item_5   NA 1.427  2.991 1.97356 -0.405    NA
Item_6   NA 0.547  2.619 1.08363 -0.958    NA
Item_7   NA 0.996  2.008 0.00761     NA    NA
Item_8   NA 0.996  1.074      NA     NA     0

Std. Errors:
         a_F1   a_F2    d_1    d_2    d_3 guess
Item_1     NA     NA 0.0834     NA     NA    NA
Item_2 0.0956     NA 0.1099     NA     NA    NA
Item_3 0.0974     NA 0.1064     NA     NA    NA
Item_4 0.0974 0.0742 0.0602     NA     NA    NA
Item_5     NA 0.0854 0.1623 0.1125 0.0634    NA
Item_6     NA 0.0531 0.1785 0.0673 0.0632    NA
Item_7     NA 0.0937 0.1100 0.0547     NA    NA
Item_8     NA 0.0937 0.0815     NA     NA    NA

GROUP PARAMETERS:
Covariance:
     F1   F2
F1 1.00 0.41
F2 0.41 1.00

Std. Errors:
       F1     F2
F1     NA 0.0175
F2 0.0175     NA

R> anova(c_mod2, c_mod)
```

Subroutine	2-factor time (s)	3-factor time (s)	4-factor time (s)
`mirt`	4.2	9.2	128.8
`ltm`	1353.1	—	—
TESTFACT	9.6	175.3	946.3
`polymirt`	117.5	172.9	202.1
`MCMCirtKd`	2150.7	2368.6	2479.5

Table 1: Average time in seconds for convergence.

```
Chi-squared difference:

X2 = 5.995 (SE = 0.076), df = 3, p = 0.1119
AIC difference = -0.005 (SE = 0.076)
BIC difference = -16.808 (SE = 0.076)
```

Comparison of these two models suggests that the added restrictions do not significantly make the model fit any worse than the less restricted one.

5. Program comparisons

As is useful with all new software, comparing results with previously established programs to check the accuracy and potential benefits of the new software is beneficial for front-end users. Here we compare the estimation results of `mirt()` and `polymirt()` with those obtained from **TESTFACT**, **MCMCpack** using `MCMCirtKd()`[3], and **ltm** using `ltm()` (however, `ltm()` cannot estimate more than two factors). Two-, three-, and four-factor models are extracted from the first simulated data set, with the three-factor solutions rotated with the varimax criterion that was used for subsequent comparisons. Note that all computations were performed on a desktop workstation with an AMD Phenom 9600 Quad-Core 2.31 GHz processor with 4-GB of RAM, and each subroutine was run five times to obtain the average computation time.

Deterministic methods were set to terminate when all parameters changed less than 0.001 between consecutive iterations, and the number of quadratures used during estimation were 20, 10, and 7, respectively. `polymirt()` was set to have 100 burn-ins, 50 cycles to find approximate starting values, and once the RM stage was implemented, the estimation was terminated when all parameters changed less than 0.001 between iterations on three consecutive cycles. For `MCMCirtKd()`, the burn-in iterations were set at 10000, with 25000 MCMC draws, thinning every 5 draws, and storing only the item parameters. Finally, for the stochastic algorithms, the first model estimated was selected for subsequent comparisons.

As can be seen in Table 1, `mirt()` and `polymirt()` were much more efficient compared to the `ltm()` and `MCMCirtKd()` functions, and while `mirt()` was consistently more efficient than **TESTFACT**, `polymirt()` did not become more efficient than **TESTFACT** until there were more than 2 factors. Also note that the estimation time for `MCMCirtKd()` was quite long, spanning between 35–41 minutes on average per model.

[3]**TESTFACT** and `MCMCirtKd()` use the traditional normal ogive item response model, so slight deviations in numerical solutions should be expected.

	Parameters			**TESTFACT**			`MCMCirtKd()`			`mirt()`			`polymirt()`		
1	**.59**	.11	.01	**.46**	.00	.14	**.44**	.05	.15	**.53**	.01	.06	**.58**	.00	.18
2	**.41**	.06	.00	**.39**	.02	.06	**.31**	.06	.06	**.38**	.01	.01	**.42**	.01	.07
3	**.62**	.29	.00	**.46**	.17	.02	**.48**	.28	.05	**.62**	.20	.02	**.62**	.23	.04
4	**.71**	.03	.00	**.49**	-.03	.04	**.52**	.02	.07	**.69**	-.07	.10	**.70**	-.05	.06
5	**.47**	.12	.01	**.41**	.07	.12	**.34**	.11	.11	**.47**	.05	.08	**.47**	.08	.14
6	**.76**	.30	.00	**.44**	.13	.03	**.64**	.31	.07	**.76**	.19	.03	**.80**	.22	.07
7	**.77**	.26	.04	**.46**	.10	.07	**.54**	.22	.11	**.75**	.16	.08	**.77**	.17	.12
8	**.63**	.19	.02	**.47**	.10	.07	**.43**	.18	.08	**.69**	.04	.10	**.62**	.13	.10
9	**.70**	.14	.00	**.49**	.03	.03	**.58**	.13	.06	**.65**	.00	.08	**.72**	.04	.05
10	**.66**	.21	.00	**.49**	.08	.03	**.55**	.19	.06	**.67**	.13	.04	**.69**	.10	.04
11	.00	**.62**	.00	.04	**.48**	.03	.10	**.51**	.00	.07	**.56**	-.05	.05	**.62**	.03
12	.00	**.76**	.12	.05	**.49**	.01	.16	**.70**	.03	.06	**.73**	.05	.09	**.74**	.02
13	.04	**.57**	.06	.11	**.46**	.01	.17	**.51**	.02	.09	**.51**	.03	.14	**.56**	.02
14	.00	**.89**	.13	.04	**.38**	.02	.16	**.64**	.04	.10	**.85**	.10	.11	**.89**	.04
15	.01	**.64**	.03	.08	**.49**	.02	.16	**.61**	.03	.09	**.60**	-.04	.11	**.67**	.03
16	.01	**.67**	.10	.07	**.48**	.07	.16	**.63**	.09	.10	**.63**	.03	.11	**.68**	.11
17	.01	**.79**	.16	.06	**.45**	.05	.19	**.78**	.09	.10	**.77**	.07	.12	**.81**	.09
18	.00	**.65**	.16	.03	**.49**	.04	.11	**.57**	.05	.07	**.64**	.07	.04	**.65**	.06
19	.00	**.55**	.35	.06	**.43**	.21	.11	**.40**	.16	.01	**.52**	.32	.08	**.56**	.29
20	.00	**.53**	.07	.05	**.46**	.02	.11	**.53**	.03	.08	**.48**	-.00	.06	**.55**	.03
21	.21	.00	**.68**	.08	.02	**.49**	.10	.03	**.51**	.12	.00	**.67**	.12	.02	**.72**
22	.16	.00	**.43**	.08	.02	**.41**	.07	.02	**.34**	.14	-.03	**.45**	.09	.02	**.45**
23	.26	.12	**.71**	.16	.10	**.45**	.22	.17	**.54**	.21	.17	**.66**	.24	.15	**.68**
24	.07	.01	**.86**	.00	.04	**.44**	.03	.04	**.45**	.03	.09	**.85**	-.00	.06	**.86**
25	.12	.00	**.55**	.03	.04	**.46**	.04	.04	**.41**	.05	.01	**.55**	.04	.04	**.56**
26	.17	.01	**.76**	.07	.04	**.47**	.11	.06	**.60**	.12	.03	**.76**	.12	.05	**.78**
27	.07	.00	**.82**	.02	.02	**.47**	.05	.02	**.66**	.01	.04	**.79**	.03	.02	**.80**
28	.04	.00	**.42**	-.01	.00	**.41**	.00	.00	**.32**	-.00	.05	**.44**	-.02	.00	**.45**
29	.19	.00	**.41**	.14	.00	**.37**	.12	.00	**.31**	.15	.04	**.38**	.16	.00	**.41**
30	.11	.05	**.73**	.06	.07	**.48**	.10	.12	**.62**	.05	.04	**.71**	.08	.11	**.74**

Table 2: Simulated parameters compared to varimax rotated solutions for **TESTFACT**, `MCMCirtKd()`, `mirt()`, `polymirt()`.

Estimation accuracy was assessed by computing the root mean-squared deviation statistic (RMSD), where lower values indicate better precision in parameter recovery, as well as by comparing the observed log-likelihood values. Table 2 compares the varimax rotated solutions for the four procedures that converged on a solution for the three-factor model. Of these four procedures, `polymirt()` had the highest log-likelihood (-30542.5), with `mirt()` (-30663.3), `MCMCirtKd()` (-31084.0), and **TESTFACT** (-33434.1) following. Additionally, `polymirt()` was the most accurate at recovering the simulated parameters (RMSD = 0.047), while `mirt()` closely followed (RMSD = 0.060). **TESTFACT** (RMSD = 0.107) and `MCMCirtKd()` (RMSD = 0.085) appeared to have suffered due to utilizing fewer quadratures per dimension, and perhaps from drawing too few MCMC values since there were convergence warnings noted for all of the estimated models.

6. Discussion

Several useful applications of the **mirt** package are possible that were not demonstrated in this article. For instance, `confmirt()` can estimate Rasch-type models stochastically by simply constraining all of the slope parameters to be equal (or exactly to 1/1.702, for the traditional Rasch model). This may be a useful strategy if the number of participants is large or the number of test items is large, since the MH-RM is well equipped to handle both of these situations. Non-linear factor combination and noncompensetory item response relationships may also be included when specifying `confmirt()` models. Finally, factor scores and information plots (and surfaces) for individual items or the whole test are available, and simulation functions are also readily at the user's disposal (see Appendix A).

As it stands, the **mirt** package appears to be a useful tool for researchers interested in exploratory and confirmatory item response models, and improves upon the overall estimation time and parameter recovery efficacy compared to various previously published software. The package is actively being developed, and some of the future developments may include:

- adding limited-information model fit statistics
- providing standard errors for EM solutions
- performing multiple-group estimation, and
- utilizing nominal and rating scale intercept methods for polytomous data

These are only a few of the potential development areas, and user interest will largely guide which features will be developed. Popular options that will be available in the **IRTPRO** software (Cai, du Toit, and Thissen 2011) also may be given precedence depending on user feedback and interests in using open-source software along with proprietary software in their item analysis work.

Acknowledgments

Special thanks to David Flora, Matthew Sigal, and two anonymous reviewers for offering helpful comments that improved the quality of this manuscript.

References

Baker FB, Kim SH (2004). *Item Response Theory: Parameter Estimation Techniques.* 2nd edition. Dekker, New York.

Bernaards CA, Jennrich RI (2005). "Gradient Projection Algorithms and Software for Arbitrary Rotation Criteria in Factor Analysis." *Educational and Psychological Measurement*, **65**, 676–696.

Birnbaum A (1968). "Some Latent Trait Models and Their Use in Inferring an Examinee's Ability." In FM Lord, MR Novick (eds.), *Statistical Theories of Mental Test Scores*, pp. 395–479. Addison-Wesley, Reading.

Bock RD, Aitkin M (1981). "Marginal Maximum Likelihood Estimation of Item Parameters: Application of an EM Algorithm." *Psychometrika*, **46**(4), 443–459.

Bock RD, Gibbons R, Muraki E (1988). "Full-Information Item Factor Analysis." *Applied Psychological Measurement*, **12**(3), 261–280.

Bock RD, Lieberman M (1970). "Fitting a Response Model for n Dichotomously Scored Items." *Psychometrika*, **35**(2), 179–197.

Bollen KA (1989). *Structural Equations with Latent Variables.* John Wiley & Sons, New York.

Bolt D (2005). "Limited and Full Information Estimation of Item Response Theory Models." In A Maydeu-Olivares, JJ McArdle (eds.), *Contemperary Psychometrics*, pp. 27–71. Lawrence Erlbaum Associates, Mahwah.

Cai L (2010a). "High-Dimensional Exploratory Item Factor Analysis by a Metropolis-Hastings Robbins-Monro Algorithm." *Psychometrika*, **75**(1), 33–57.

Cai L (2010b). "Metropolis-Hastings Robbins-Monro Algorithm for Confirmatory Item Factor Analysis." *Journal of Educational and Behavioral Statistics*, **35**(3), 307–335.

Cai L (2010c). "A Two-Tier Full-Information Item Factor Analysis Model with Applications." *Psychometrika*, **75**(4), 581–612.

Cai L, du Toit SHC, Thissen D (2011). ***IRTPRO**: Flexible, Multidimensional, Multiple Categorical IRT Modeling.* Scientific Software International.

Casella G, George EI (1992). "Explaining the Gibbs Sampler." *The American Statistician*, **46**, 167–174.

Chen WH, Thissen D (1997). "Local Dependence Indices for Item Pairs Using Item Response Theory." *Journal of Educational and Behavioral Statistics*, **22**, 265–289.

Dempster AP, Laird NM, Rubin DB (1977). "Maximum Likelihood From Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society B*, **39**(1), 1–38.

Edwards MC (2010). "A Markov Chain Monte Carlo Approach to Confirmatory Item Factor Analysis." *Psychometrika*, **75**(3), 474–497.

Fraser C (1998). ***NOHARM**: A Fortran Program for Fitting Unidimensional and Multidimensional Normal Ogive Models in Latent Trait Theory.* Armidale, Australia. The University of New England, Center for Behavioral Studies.

Gibbons RD, Darrell RB, Hedeker D, Weiss DJ, Segawa E, Bhaumik DK, Kupfer DJ, Frank E, Grochocinski VJ, Stover A (2007). "Full-Information Item Bifactor Analysis of Graded Response Data." *Applied Psychological Measurement*, **31**(1), 4–19.

Gibbons RD, Hedeker DR (1992). "Full-Information Item Bi-Factor Analysis." *Psychometrika*, **57**, 423–436.

Hastings WK (1970). "Monte Carlo Simulation Methods Using Markov Chains and Their Applications." *Biometrika*, **57**, 97–109.

Holzinger KJ, Swineford F (1937). "The Bi-Factor Method." *Psychometrika*, **2**(1), 41–54.

Mair P, Hatzinger R (2007). "Extended Rasch Modeling: The **eRm** Package for the Application of IRT Models in R." *Journal of Statistical Software*, **20**(9), 1–20. URL http://www.jstatsoft.org/v20/i09/.

Martin AD, Quinn KM, Park JH (2011). "**MCMCpack**: Markov Chain Monte Carlo in R." *Journal of Statistical Software*, **42**(9), 1–21. URL http://www.jstatsoft.org/v42/i09/.

McDonald RP (1999). *Test Theory: A Unified Treatment*. Lawrence Erlbaum Associates, Mahawah.

Metropolis N, Rosenbluth AW, Teller AH, Teller E (1953). "Equations of State Space Calculations by Fast Computing Machines." *Journal of Chemical Physics*, **21**, 1087–1091.

Muraki E, Carlson EB (1995). "Full-Information Factor Analysis for Polytomous Item Responses." *Applied Psychological Measurement*, **19**, 73–90.

Muthén LK, Muthén BO (2008). ***Mplus** (Version 5.0)*. Muthén & Muthén, Los Angeles.

R Development Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Reckase MD (2009). *Multidimensional Item Response Theory*. Springer-Verlag, New York.

Reckase MD, McKinley R (1991). "The Discrimination Power of Items that Measure More than One Dimension." *Applied Psychological Measurement*, **15**, 361–373.

Rizopoulos D (2006). "**ltm**: An R Package for Latent Variable Modeling and Item Response Theory Analysis." *Journal of Statistical Software*, **17**(5), 1–25. URL http://www.jstatsoft.org/v17/i05.

Robbins H, Monro S (1951). "A Stochastic Approximation Method." *The Annals of Mathematical Statistics*, **22**, 400–407.

Samejima F (1969). "Estimation of Latent Ability Using a Response Pattern of Graded Scores." *Psychometrika Monographs*, **34**(4).

Schilling S, Bock RD (2005). "High-Dimensional Maximum Marginal Likelihood Item Factor Analysis by Adaptive Quadrature." *Psychometrika*, **70**, 533–555.

Sheng Y (2010). "Bayesian Estimation of MIRT Models with General and Specifc Latent Traits in MATLAB." *Journal of Statistical Software*, **34**(3), 1–27. URL http://www.jstatsoft.org/v34/i03/.

Thissen D, Wainer H (eds.) (2001). *Test Scoring*. Lawrence Erlbaum Associates, Mahwah.

Thurstone LL (1947). *Multiple Factor Analysis*. University of Chicago Press, Chicago.

Wainer H, Bradlow ET, Wang X (2007). *Testlet Response Theory and Its Applications*. Cambridge University Press, New York.

Weeks JP (2010). "**plink**: An R Package for Linking Mixed-Format Tests Using IRT-Based Methods." *Journal of Statistical Software*, **35**(12), 1–33. URL http://www.jstatsoft.org/v35/i12/.

Wirth RJ, Edwards MC (2007). "Item Factor Analysis: Current Approaches and Future Directions." *Psychological Methods*, **12**(1), 58–79.

Wood R, Wilson DT, Gibbons RD, Schilling SG, Muraki E, Bock RD (2003). ***TESTFACT** 4 for Windows: Test Scoring, Item Statistics, and Full-Information Item Factor Analysis.* Scientific Software International.

Yao L (2008). ***BMIRT**: Bayesian Multivariate Item Response Theory (Version 1.0).* CTB/McGraw-Hill, Monterey.

A. Additional features

Additional features in `plot()`, `itemplot()` and `fscores()` are illustrated in the following.

Unidimensional and two-factor test information plots for the LSAT7 data can be produced with the `plot()` method (see Figure 1).

```
R> plot(mod1)
R> plot(mod2)
```

Probability plots for each LSAT7 item are generated with the `itemplot()` function (see Figure 2).

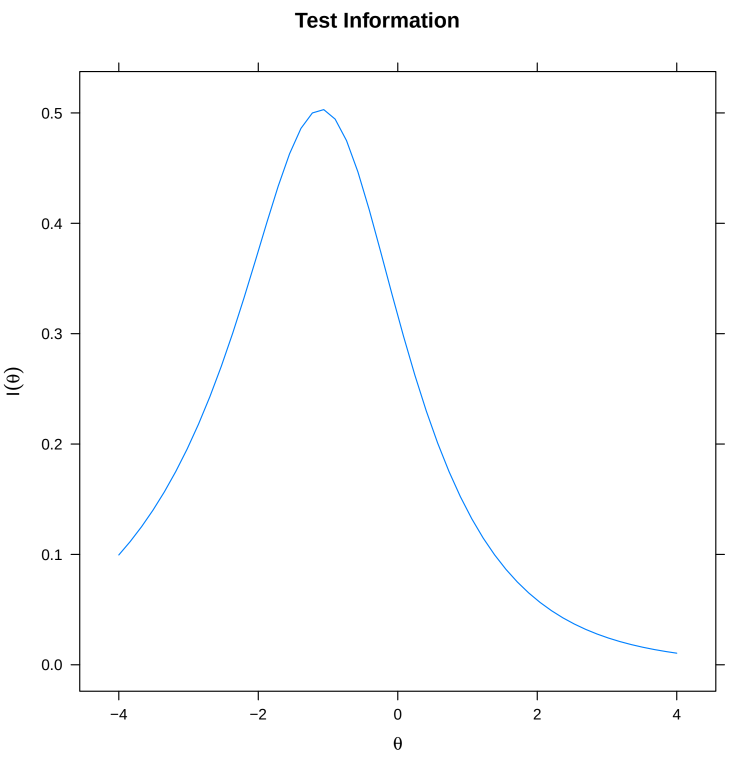
```
itemplot(mod1, combine = 5, auto.key = list(space = 'right'))
```

The following code returns either a table of EAP or MAP factor scores of each unique response pattern, or appends the factor scores to the last column of the input data matrix for each response pattern.

```
R> fscores(mod1)

Method:  EAP

      Item.1 Item.2 Item.3 Item.4 Item.5 Freq         F_1    SE_F_1
 [1,]      0      0      0      0      0   12 -1.87028659 0.6900538
 [2,]      0      0      0      0      1   19 -1.52611274 0.6734105
 [3,]      0      0      0      1      0    1 -1.51372763 0.6729447
 [4,]      0      0      0      1      1    7 -1.18234637 0.6656759
 [5,]      0      0      1      0      0    3 -1.10291088 0.6655960
 [6,]      0      0      1      0      1   19 -0.77181736 0.6726494
 [7,]      0      0      1      1      0    3 -0.75944267 0.6731393
 [8,]      0      0      1      1      1   17 -0.41453481 0.6926378
 [9,]      0      1      0      0      0   10 -1.37398604 0.6685939
[10,]      0      1      0      0      1    5 -1.04466802 0.6659735
[11,]      0      1      0      1      0    3 -1.03254430 0.6660987
[12,]      0      1      0      1      1    7 -0.69967712 0.6757235
[13,]      0      1      1      0      0    7 -0.61731385 0.6798590
[14,]      0      1      1      0      1   23 -0.26302205 0.7042673
[15,]      0      1      1      1      0    8 -0.24944505 0.7053837
[16,]      0      1      1      1      1   28  0.13766313 0.7409730
[17,]      1      0      0      0      0    7 -1.41222110 0.6696109
[18,]      1      0      0      0      1   39 -1.08252979 0.6656860
[19,]      1      0      0      1      0   11 -1.07041745 0.6657609
[20,]      1      0      0      1      1   34 -0.73857313 0.6740009
[21,]      1      0      1      0      0   14 -0.65666864 0.6778019
[22,]      1      0      1      0      1   51 -0.30517792 0.7008748
[23,]      1      0      1      1      0   15 -0.29173219 0.7019445
[24,]      1      0      1      1      1   90  0.09106181 0.7363931
[25,]      1      1      0      0      0    6 -0.93235512 0.6677492
```



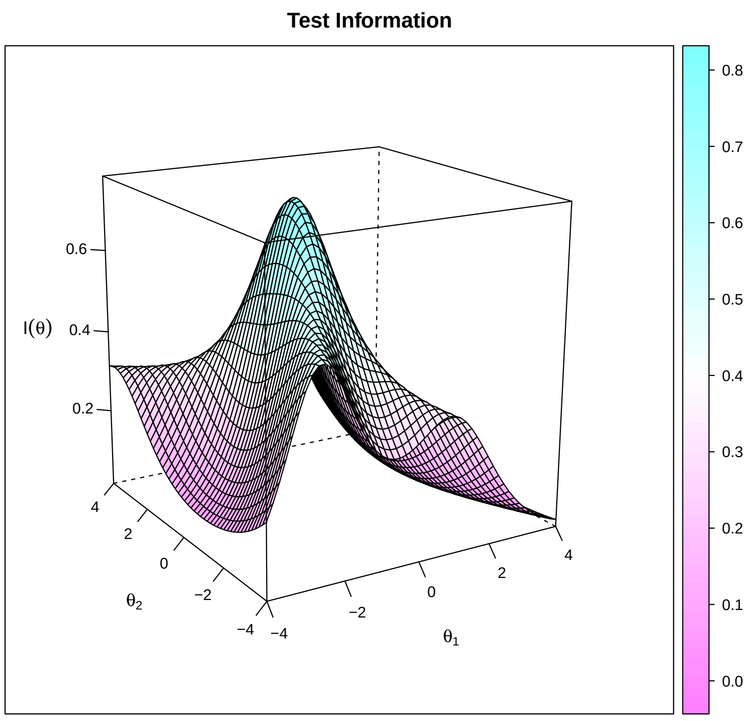


Figure 1: Unidimensional and two-factor test information plots for the LSAT7 data.

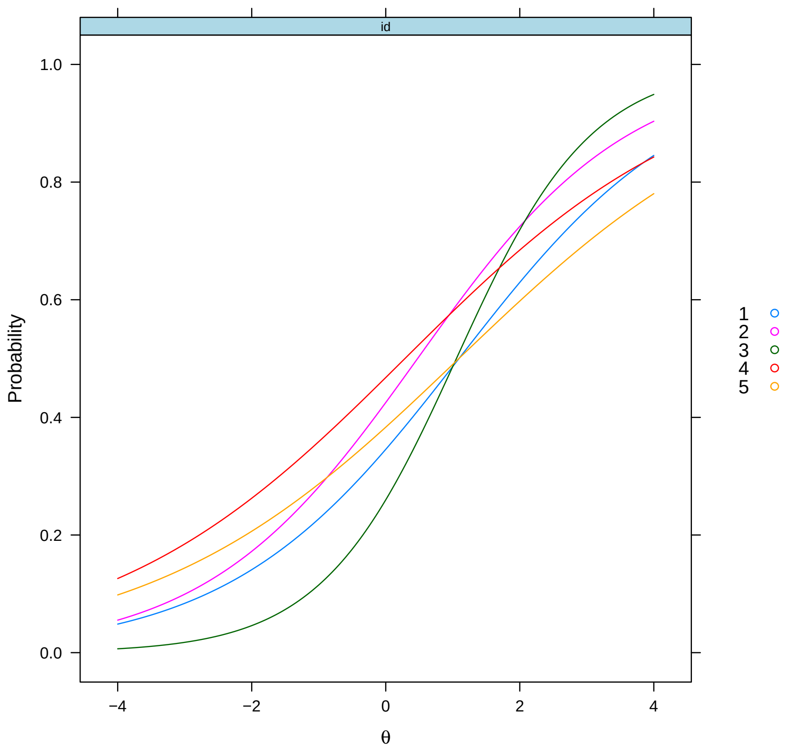


Figure 2: Combined probability plot for each LSAT7 item.

```
[26,]      1      1      0      0      1   25 -0.59601605 0.6810327
[27,]      1      1      0      1      0    7 -0.58332684 0.6817517
[28,]      1      1      0      1      1   35 -0.22649542 0.7072960
[29,]      1      1      1      0      0   18 -0.13580459 0.7151417
[30,]      1      1      1      0      1  136  0.26346482 0.7535721
[31,]      1      1      1      1      0   32  0.27901726 0.7551477
[32,]      1      1      1      1      1  308  0.72759057 0.8007276
```

```
R> dataWithFS <- fscores(mod1, full.scores = TRUE)
```

B. Simulation parameters

Simulation parameters for `confmirt()` .

```
R> a <- matrix(c(1.5, NA, 0.5, NA, 1, NA, 1, 0.5, NA, 1.5, NA, 0.5,
+    NA, 1, NA, 1), ncol = 2, byrow = TRUE)
R> d <- matrix(c(-1, NA, NA, -1.5, NA, NA, 1.5, NA, NA, 0, NA, NA,
+    3, 2, -0.5, 2.5, 1, -1, 2, 0, NA, 1, NA, NA), ncol = 3, byrow = TRUE)
R> sigma <- diag(2)
R> sigma[1, 2] <- sigma[2, 1] <- 0.4
R> simdata2 <- simdata(a, d, 2000, sigma)
```

Affiliation:

R. Philip Chalmers
Department of Psychology
York University
4700 Keele St.
Toronto, ON, M3J 1P3, Canada
Email: rphilip.chalmers@gmail.com

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/

Volume 48, Issue 6 *Submitted:* 2011-08-24
May 2012 *Accepted:* 2012-02-08