Unidimensional and Multidimensional IRT Modeling with the `mirt` Package

Phil Chalmers

York University

February 18, 2013

Introduction

This presentation focuses on unidimensional and multidimensional item response theory (UIRT and MIRT, respectively) models that can be estimated with the `mirt` (Chalmers, 2012) package. In general, I will go over:

- What IRT is, why it exists, and how it relates to other latent variable methods such as factor analysis
- Several types of IRT models and how these can be generalized to more than one dimension
- How to fit UIRT and MIRT models to psychological test data with the `mirt` package
- Useful model comparison techniques, computing latent trait scores and item/person fit statistics, plotting item and test probability curves and information functions, and
- (time permitting) Explore some more advanced methods such as multiple group analysis for detecting DIF, user defined prior parameter distributions and starting values, linear parameter constraints, Wald tests, etc.

Classical Test Theory

Classical test theory was largely developed by Spearman, Thurstone, Kuder, Guttman, and Cronbach, as well as a few others. In general to determine the properties of a scale the following aspects were studied (almost entirely by linear regression theory):

1) Estimating the *global* reliability of a test based on how homogeneous the items are with each other (α, split-half), and using this to define the *global* standard error of measurement
2) Use the total score of a test as an estimate of ability/'True score' ($X = T + E$) and studying how each individual item relates to this total score
3) Determining the number of linearly related latent factors are manifested in a test (via factor analysis or structure equation modeling), and try to reduce the number of factors down to 1

Classical Test Theory Problems

- Standard error applies to everyone in the population (10 ± 2, 5 ± 2)
- To compare tests to each other forms must be *parallel* (equal item difficulties, same number of items, etc.)
- Individual scores are understood by comparing the person to the group (make total into z or T-scores)
- Mixed item formats are difficult to compare (multiple choice vs true-false) and become ambiguous when combined for a total score
- Factor analysis on binary items leads to “difficulty” artifact dimensions
- Change scores cannot be meaningfully compared when initial score levels differ

Item Response Theory

- Item response theory (IRT) is a set of latent variable techniques specifically designed to model the interaction between a subject's 'ability' and item level stimuli (difficulty, guessing, etc.)
- Focus is on the pattern of responses rather than on composite variables and linear regression theory, and emphasises how responses can be thought of in probabilistic terms
- Much larger emphases on the error of measurement for *each* test subject rather than a global index of reliability/measurement error
- Widely used in educational and psychological research to study latent variable constructs other than ability (e.g., depression, personality, motivation)

Most common IRT models are still unidimensional, meaning they relate the items to only one latent trait, although multidimensional IRT models are becoming more popular

Unidimensional IRT models (dichotomous)

Traditional IRT models were developed for modeling how a subject's 'ability' (θ) was related to answering a test item correctly ($0 =$ incorrect, $1 =$ correct) given item level proprieties.

$$P(x = 1; \theta, a, d) = \frac{1}{1 + \exp\left(-D(a\theta + d)\right)}$$

This equation represents the 2 parameter logistic model (2PL). The D parameter is a constant used to transform the overall metric to make the model closer to traditional factor analysis, commonly taken to be 1.702.

- Given some ability level, θ, the probability of correct endorsement is related to the item easiness (d) and it's slope/discrimination (a). It may be easier to understand these relationships in the canonical form: $\log(P) \approx a\theta + d$
- This model is tied very closely to factor analysis on tetrachoric correlations, and has an analogous relationship to multiple factor analysis when the number of factors is greater than one (i.e., multidimensional)

Unidimensional plots (2PL)

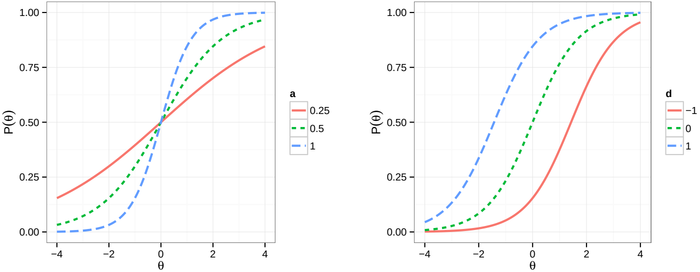


Figure: Item response curves when varying the slope and intercept parameters in the 2PL model (not generated from `mirt`)

Unidimensional IRT models (dichotomous, cont.)

Further generalization of the 2PL model are also possible to accommodate for other psychological phenomenon such as guessing or ceiling effects. For example,

$$P(x = 1; \theta, a, d, \gamma, \delta) = \gamma + \frac{(\delta - \gamma)}{1 + \exp\left(-1.702(a\theta + d)\right)}$$

This is the (maybe not so popular, but still pretty cool) four parameter logistic model, which when specific constraints are applied reduces to the 3PL, 2PL, 1PL, and Rasch model.

- Given some ability level, θ, the probability of correct endorsement is related to the item easiness (d), discrimination (a), probability of randomly guessing (γ), and probability of randomly answering incorrectly (δ).
- For psychological questionnaires the lower and upper bounds often have no rational and are taken to be 0 and 1, respectively (though in clinical instruments they may be justified).

Unidimensional plots (4PL)

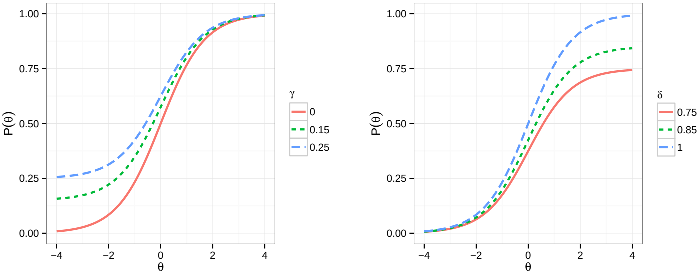


Figure: Item response curves when varying the lower and upper bound parameters in the 4PL model (not generated from `mirt`)

Unidimensional IRT models (polytomous)

Several different kinds of polytomous item response models exist for ordinal, rating scale, generalized partial credit, and nominal models; all of which extend to the multidimensional case (some of which require some initially counterintuitive parameterizations). Likert scales, for example, are often modeled by ordinal or rating scale models. The ordinal/graded response model can be expressed as:

$$P(x_k = k; \theta, \phi) = P(x \geq k) - P(x \geq k + 1)$$

For the generalized partial credit model the d_k values are treated as fixed and ordered values from $0 : (k - 1)$.

$$P(x = k; \theta, \psi) = \frac{exp(-1.702[ak_k(a\theta) + d_k])}{\sum_{j=1}^{k} exp(-1.702[ak_k(a\theta) + d_k])}$$

Unidimensional plots (polytomous)

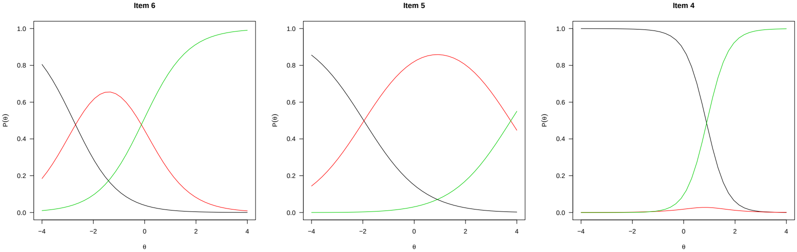


Figure: Probability curves for ordinal (left), generalized partial credit (middle), and nominal (right) response models

Item and test information

Item and test information are very important concepts in IRT and form the building blocks of more advanced applications such as computerized adaptive testing (CAT). The information in a test depends on the items used **as well as the ability of the subject**, and is inversely related to *reliability*. IRT advances the concept of reliability by treating it as a function of the θ values

- For example, easy items and tests tend to tell us very little about individuals in the upper end of the θ distribution ($\theta_{Einstein}$ v.s. $\theta_{Hawking}$) but can tell us something about lower ability subjects (whether $\theta_{Larry} < \theta_{Curly} < \theta_{Moe}$).
- Formally this information function (dependent on θ) is defined as:

$$I(\theta) = \sum_{k=1} \left(\frac{(\partial P/\partial\theta)^2}{P} - \partial^2 P/\partial\theta \right)$$

- *Test information* is simply the sum over each item information function $T(\theta) = \sum_{i=1} I_i(\theta)$. CAT applications often stop when the information reaches a pre-specified tolerance (since $\text{SE}(\theta) = \sqrt{T(\theta)^{-1}}$). These ideas also readily generalize to multiple latent traits

Ability estimation

Three algorithms are typically used to obtain estimates of latent trait values and their standard errors:

1) Maximum likelihood (ML) – Maximize likelihood vector w.r.t. θ directly with iterative methods. Doesn't allow for all/none patterns
2) Maximum a posteriori (MAP) – Given a prior (typically [multivariate] normal) maximize the posterior distribution. Requires iterative methods for each response pattern but works for all patterns
3) Expected a posteriori (EAP) – Similar to MAP but is not iterative and often a consequence of the estimation process (mean estimate rather than mode). Most often used method
4) Weighted Likelihood Estimation (WLE) – An iterative estimate of the latent trait that weighs the scores based on how much information is available from the test (often falls between ML and MAP)

Multidimensional IRT models

Multidimensional IRT models replace the single θ and a values with vectors $\boldsymbol{\theta}$ and $\mathbf{a}$, respectively. This is analogous to the transition from zero-order regression to multiple regression (expect that the predictors are latent and non-linear).

$$P(x = 1; \boldsymbol{\theta}, \mathbf{a}, d, \gamma, \delta) = \gamma + \frac{(\delta - \gamma)}{1 + \exp\left[-1.702(\mathbf{a}'\boldsymbol{\theta} + d)\right]}.$$

This model has a very intimate relationship to nonlinear factor analysis when $\gamma = 0$ and $\delta = 1$, (since $log(P) \approx \mathbf{a}'\boldsymbol{\theta} + d$) and is often called a 'compensatory' model for the relationships between latent trait scores.

- Similar relationships exists for the generalized partial credit, graded, and nominal models, but other special types of models that don't follow these trends (e.g., partially compensatory, polynomial/exponential related traits) are also possible.

Multidimensional plots

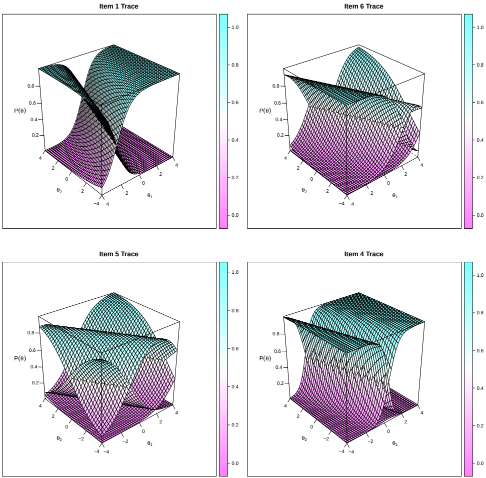


Figure: Probability curves for multidimensional 2PL and ordinal (top), generalized partial credit and nominal models (bottom)

Model estimation

IRT item parameters are typically estimated by maximizing the observed likelihood

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

- Maximizing the above equation directly quickly becomes infeasible due to the number of parameters estimated
- Instead an EM algorithm is often employed to capitalize on a more manageable complete-data likelihood (creating artificial tables of number of participants with given response patterns)
- Effectively this approach lessens the problem of maximizing all the parameters at each iteration, but the integrals must still be evaluated

Unfortunately . . .

Every new θ estimated requires a new integral to be evaluated in the observed likelihood.

- The difficult task is to evaluate the likelihood numerically, which requires integration by quadrature (e.g., Gauss-Hermite) or simulation methods
- Quadrature techniques often become intractable as the dimensions increase since the number of quadratures required increases exponentially
- Bayesian methods have been used to circumvent this integration problem at the cost of longer estimation times and often high computation demand

Estimation (cont.)

An alternative approach is to capitalize on the complete-data likelihood function directly

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

- What is required here is that we obtain 'known' values for $\boldsymbol{\theta}$ and maximize this function instead
- The Metropolis-Hastings Robbins-Monro (MH-RM) algorithm works well in this situation and is surprisingly fast and accurate
- MH sampler to obtain $\boldsymbol{\theta}$ values, treat values as 'known' and update parameters using standard numerical optimization methods (e.g., Newton-Raphson), and use Robbins-Monro method help remove the sampling error borne from the MH draws

mirt package tip

I recommend using the MH-RM over the EM when the number of dimensions in the model becomes higher than 3–4

mirt package

mirt package

Why the mirt package?

1) Multidimensional IRT functions in R offered limited features, were slow, and sometimes computationally demanding (e.g., ltm, MCMCpack)
2) Wanted an open source version of TESTFACT and POLYFACT which would easily integrate with useful R packages (e.g., plink, GPArotation)
3) Also wanted to utilize the MH-RM algorithm (Cai, 2010) for higher dimensional and confirmatory IRT models (analogous to confirmatory factor analysis in SEM)
4) Wanted to fit more general item response models (e.g., nominal, generalized partial credit, partially compensatory, polynomial related traits, etc.)
5) Wanted multiple group estimation, which is important for testing the bias in testing instruments. Existed in proprietary software (even then, only in a select few) but couldn't work for MIRT models
6) Finally, for modelling fixed and random predictor variables directly in IRT models

Functions

The mirt package consists of 5 estimation functions: mirt(), bfactor(), confmirt(), multipleGroup(), and mixedmirt(). All of these function can be used to model any mixture of dichotomous and polytomous items.

- mirt() uses a fixed quadrature estimation method (Bock & Aitkin, 1981) for obtaining ML parameter estimates with the EM algorithm. The syntax used is similar to the standard factor analysis routines in R, but also allows for confmirt.model() defined objects
- bfactor() uses dimension reduction algorithm for confirmatory bi-factor models described by Gibbons, Darell, Hedeker, et al. (2007). These have the benefit of remaining computationally efficient regardless of the number of specific factors
- confmirt() uses the MH-RM algorithm for exploratory and confirmatory IRT models, which may also include non-compensatory item types and polynomial factor relationships
- multipleGroup() uses the MH-RM or EM algorithm to perform multiple group estimation useful for testing the invariance of parameters between potentially heterogeneous groups
- mixedmirt() uses the MH-RMalgorithm to estimate fixed or random effect covariates at the item or person level (e.g., LLTM)

Functions (cont.)

Some useful generic functions which work on the returned estimated objects:

- `coef()` and `summary()` – extract unstandardized and standardized (i.e., factor loadings) coefficients, respectively
- `plot()` – two- and three-dimensional probability and information plots for item bundles
- `anova()` – comparison between nested models with χ^2, AIC, BIC, etc.
- `residuals()` and `fitted()` – linear dependence or pattern based residuals
- `itemplot()` – plots individual item response curves
- `fscores()` – compute EAP, MAP, WLE, or ML factor scores
- `itemfit()` – Z, χ^2, infit, and outfit statistics to judge item fit
- `personfit()` – Z, infit, and outfit for detecting person misfit
- `imputeMissing()` – impute plausible responses given $\hat{\theta}$
- `read.mirt()` – convert models to objects usable by the `plink` package

Possible MIRT models

From the `mirt` documentation:

> **itemtype**
>
> type of items to be modeled, declared as a vector for each item or a single value which will be repeated globally. The NULL default assumes that the items follow a graded or 2PL structure, however they may be changed to the following: ’Rasch’, ’1PL’, ’2PL’, ’3PL’, ’3PLu’, ’4PL’, ’graded’, ’grsm’, ’gpcm’, ’rsm’, ’nominal’, ’mcm’, ’PC2PL’, and ’PC3PL’, for the Rasch/partial credit, 1 and 2 parameter logistic, 3 parameter logistic (lower asymptote and upper), 4 parameter logistic, graded response model, rating scale graded response model, Rasch rating scale, generalized partial credit model, nominal model, multiple choice model, and 2-3PL partially compensatory model, respectively

See `?mirt` for more details.

Running example

To demonstrate some of the features in mirt I've constructed a simple dataset of 6 items consisting of 2PL, ordinal, gpcm, and nominal item models with an orthogonal bi-factor structure (one general factor that affects all items + specific item factors that form a Thurstonian 'simple structure'). This dataset was used to generate the previous figures as well and came from the mirt function simdata().

```
> cat(itemtype)

## 2PL 2PL 2PL nominal gpcm graded

> head(dat)

##      Item_1 Item_2 Item_3 Item_4 Item_5 Item_6
## [1,]      0      0      0      1      1      2
## [2,]      0      0      0      1      1      1
## [3,]      0      1      0      1      1      2
## [4,]      0      0      0      1      1      2
## [5,]      0      1      0      1      1      2
## [6,]      1      1      0      1      0      3
```

mirt() estimation

```
> # one factor
> mixedmod <- mirt(dat, 1, itemtype = itemtype)
> # two factor (exploratory)
> mixedmod2 <- mirt(dat, 2, itemtype = itemtype)
> mixedmod

##
## Call:
## mirt(data = dat, model = 1, itemtype = itemtype)
##
## Full-information item factor analysis with 1 factors
## Converged in 14 iterations with 40 quadrature.
## Log-likelihood = -13986
## AIC = 28004
## AICc = 28009
## BIC = 28105
## SABIC = 28054
## G^2 = 266.3, df = 98, p = 0
## TLI = 0.896, RMSEA = 0.021
```

Estimation times

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

Table: Estimation times in seconds for three factor population model. See Chalmers (2012) for more detail.

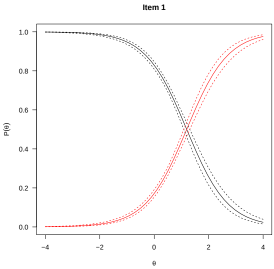
summary()

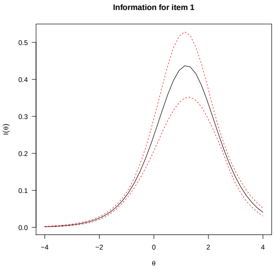
```
> summary(mixedmod2, rotate = "oblimin", suppress = 0.3)

##
## Rotation:  oblimin
##
## Rotated factor loadings:
##
##          F_1    F_2    h2
## Item_1 0.654     NA 0.485
## Item_2 0.532     NA 0.261
## Item_3 0.677     NA 0.439
## Item_4    NA -0.526 0.391
## Item_5    NA -0.585 0.281
## Item_6    NA -0.588 0.365
##
## Rotated SS loadings:  1.197 0.97
##
## Factor correlations:
##
##        F_1    F_2
## F_1  1.000 -0.642
## F_2 -0.642  1.000
```

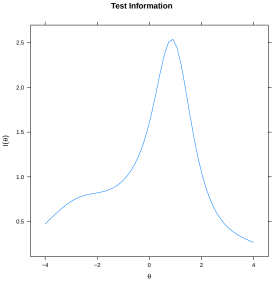
plot() and itemplot()

Confidence envelopes can be included if the information matrix was computed.

```
> itemplot(mixedmod, item = 1, CE = TRU)
> itemplot(mixedmod, item = 1, type = "info", CE = TRUE)
> plot(mixedmod)
```


Item 1


Information for item 1


Test Information

fscores()

EAP, MAP, WLE, and ML factor scores available for all estimated objects.

```
> tabscores <- fscores(mixedmod)

##
## Method:  EAP
##
## Empirical Reliability:
##     F1
## 0.6012

> head(tabscores)

##      Item_1 Item_2 Item_3 Item_4 Item_5 Item_6 Freq       F1  SE_F1
## [1,]      0      0      0      1      1      2  102 -1.03790 0.6900
## [2,]      0      0      0      1      1      1   14 -1.50165 0.7437
## [3,]      0      1      0      1      1      2  474 -0.64624 0.6647
## [4,]      1      1      0      1      0      3   11 -0.01693 0.6202
## [5,]      0      1      1      1      1      3  344  0.21448 0.5987
## [6,]      1      1      1      3      1      3  240  1.50989 0.6160
```

residuals()

```
> residuals(mixedmod2)

## LD matrix (lower triangle) and standardized values:

##         Item_1 Item_2 Item_3 Item_4 Item_5 Item_6
## Item_1      NA  0.007  0.006  0.009  0.009  0.007
## Item_2 -0.171      NA  0.007  0.004  0.017  0.015
## Item_3  0.137  0.172      NA  0.004  0.014  0.002
## Item_4  0.308 -0.055  0.071      NA  0.008  0.007
## Item_5 -0.342 -1.171  0.767 -0.259      NA  0.019
## Item_6  0.209  0.862  0.025  0.196  1.483      NA

>
> # for pattern based residuals
> head(patresid <- residuals(mixedmod, restype = "exp"))

##   Item_1 Item_2 Item_3 Item_4 Item_5 Item_6 Freq    exp    res
## 1      0      0      0      1      1      2  102  80.69  2.463
## 2      0      0      0      1      1      1   14  19.44 -1.201
## 3      0      1      0      1      1      2  474 498.95 -0.924
## 4      1      1      0      1      0      3   11  13.18 -0.572
## 5      0      1      1      1      1      3  344 385.32 -1.941
## 6      1      1      1      3      1      3  240 225.78  1.084
```

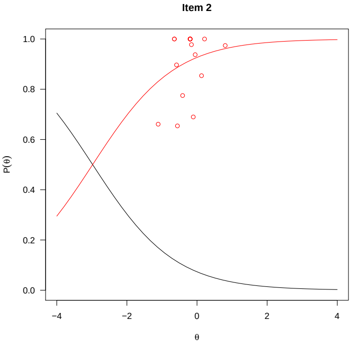
itemfit() and personfit()

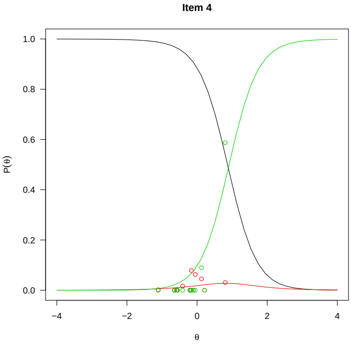
Values for detecting peculiar response patterns (e.g., someone answers all the hard questions right but easy ones wrong). Same for items, but could also also calculate a χ^2 test and plot the fitted values.

```
> pfit <- personfit(mixedmod)
> print(pfit[1:3, ])

##   Item_1 Item_2 Item_3 Item_4 Item_5 Item_6      Zh
## 1      0      0      0      1      1      2 -0.1218
## 2      0      0      0      1      1      1 -0.8980
## 3      0      1      0      1      1      2  0.9904

> ifit <- itemfit(mixedmod, X2 = TRUE)
> print(ifit[1:3, ])

##     item      Zh df     X2
## 1 Item_1  6.0559 18  93.71
## 2 Item_2  0.5401 18  43.83
## 3 Item_3 15.1659 18 286.73
```

Empirical plot

```
> itemfit(mixedmod, empirical.plot = 2)
> itemfit(mixedmod, empirical.plot = 4)
```





bfactor() estimation

```
> # specify where the specific factor load
> sp <- c(1, 1, 1, 2, 2, 2)
> bfactor.mod <- bfactor(dat, sp, itemtype, SE = TRUE)
> coef(bfactor.mod)
```

```
## $Item_1
##         a1    a2 a3      d  g  u
## pars 0.870 0.438  0 -1.023  0  1
## SE   0.034 0.032 NA  0.032 NA NA
##
## $Item_2
##         a1    a2 a3     d  g  u
## pars 0.478 0.368  0 1.562  0  1
## SE   0.039 0.038 NA 0.043 NA NA
##
## $Item_3
##         a1    a2 a3     d  g  u
## pars 0.742 0.515  0 0.022  0  1
## SE   0.028 0.025 NA 0.023 NA NA
##
## $Item_4
##         a1 a2    a3 ak0   ak1 ak2 d0     d1    d2
## pars 0.756  0 0.349   0 0.902   2  0 -0.862 1.423
## SE   0.027 NA 0.019  NA 0.127  NA NA  0.095 0.049
##
```

confmirt() estimation

This estimation function requires that a structural model be defined (models can also be passed to `mirt()`, however `confmirt()` can be more accurate and faster in higher dimensions)

```
> model <- confmirt.model()
+      G = 1-6
+     S1 = 1-3
+     S2 = 4-6

> conf.mod <- confmirt(dat, model, itemtype = itemtype, verbose = FALSE)
> anova(mixedmod, conf.mod)

##
## Model 1: mirt(data = dat, model = 1, itemtype = itemtype)
## Model 2: confmirt(data = dat, model = model, itemtype = itemtype, verbose =

##   Df   AIC  AICc   BIC SABIC logLik     X2 df p
## 1 98 28004 28009 28105 28054 -13986
## 2 95 27920 27928 28040 27979 -13941 89.712  3 0
```

Advanced features

Advanced features

Does anybody have the time? Or the patience? Or, preferably, *both*?

Customizing values and estimation

I've centered several methods for constraints, starting/fixed values, prior distributions, etc., on the idea of returning a `values` index to see how `mirt` codes the parameters. The data frame returned can then be modified and input back into the function, or users can observe what the parameter numbers are and apply linear constraints or prior parameter distributions.

```
> values <- mirt(dat, model, itemtype, pars = "values")
> head(values)

##   group   item name parnum  value lbound ubound   est
## 1   all Item_1   a1      1  0.500   -Inf    Inf  TRUE
## 2   all Item_1   a2      2  0.500   -Inf    Inf  TRUE
## 3   all Item_1   a3      3  0.000   -Inf    Inf FALSE
## 4   all Item_1    d      4 -1.033   -Inf    Inf  TRUE
## 5   all Item_1    g      5  0.000    0.0    0.5 FALSE
## 6   all Item_1    u      6  1.000    0.5    1.0 FALSE

> # change start value
> values[1, 5] <- 1
> newmod <- mirt(dat, model, itemtype, pars = values)
```

Constraints and prior distributions

Once the parameter index has been obtained users can use this information to impose equality constraints or give prior distributions to help control unstable parameters.

```
> #set first two slopes equal
> constrmod <- mirt(dat, model, itemtype,
+     constrain = list(c(1,7)))
>
> #normal prior on first intercept (N ~ (0,2))
> priormod <- mirt(dat, model, itemtype,
+     parprior = list(c(4, ’norm’, 0, 2)))
```

Multiple group estimation

Multiple group analysis (MGA) takes into account empirical grouping clusters that are thought to behave differently to the response data. For instance, items may be more difficult for one group or another, may have unequal slopes, etc., and these play a key role in determining the 'fairness' of a test.

- Two extremes of MGA are that all the parameters are equal across groups (equivalent to fitting any of the previous methods to all the data while ignoring group membership), or that all groups are completely independent (equivalent to sub-setting the data by group and estimating independent models)
- MGA becomes useful when models lie somewhere in the middle of these extremes, where we seek for a simpler model than strict independence while being mindful of population differences

Multiple group estimation (cont.)

The `multipleGroup()` function begins at the strict independence end of MGA. Although it's entirely possible to declare values manually I've included a few common across group constraints such as `slopes`, `intercepts`, `free_means`, etc., that can be passed to an optional `invariance` input.

```
> #strictly independent model
> levels(group)

## [1] "D1" "D2"

> # model can also be a confmirt.model() object
> mg1 <- multipleGroup(dat, model = 1, group = group,
+     method = 'EM', verbose = FALSE)
```

Multiple group estimation (cont.)

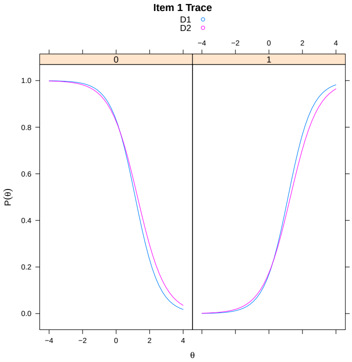
Equal slopes across groups (Wald test may be useful here too). *Note*: can use previously estimated models to give the current model free parameters better starting values.

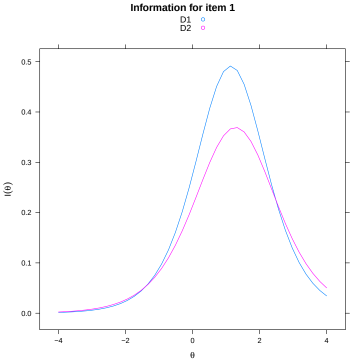
```
> mg2 <- multipleGroup(dat, model = 1, group = group,
+      prev.mod = mg1, invariance = 'slopes', method = 'EM',
+      verbose = FALSE)
> anova(mg2, mg1)

##
## Model 1: multipleGroup(data = dat, model = 1, group = group,
##     method = "EM", prev.mod = mg1, verbose = FALSE)
## Model 2: multipleGroup(data = dat, model = 1, group = group,
##     verbose = FALSE)

##    Df   AIC  AICc   BIC SABIC logLik    X2 df     p
## 1 169 27855 27890 27508 27683 -13982
## 2 163 27859 27888 27550 27706 -13978 7.768  6 0.256

> #models not sig diff, equal slopes accross
> #groups probably kool
```

Multiple group estimation itemplots

Superimposed item trace and information plots with each group. Also available for polytomous and two factor IRT models.

```
> itemplot(mg1, item = 1)
> itemplot(mg1, item = 1, type = "info")
```





mixedmirt()

The purpose of mixedmirt() is to include continuous or categorical item and person predictors into the model directly. An example of including a fixed effect predictor into the model at the person level would be the inclusion of 'Gender', where an indicator coding is used to change the expected probability to:

$$P(x = 1; \boldsymbol{\theta}, \Psi, \beta_{male}) = \gamma + \frac{(\delta - \gamma)}{1 + \exp\left[-1.702(\mathbf{a}'\boldsymbol{\theta} + d + \beta_{male} Gender)\right]}.$$

Constraining the structure of the intercept variables is also possible and is analogous to the LLTM model (Fisher, 1983), though using this approach it is not limited to Rasch models. Currently the function only supports the inclusion of fixed effect predictors at the item and person level, though support random effects are being developed. See ?mixedmirt for examples.

Future developments

This package is geared towards making complex IRT modeling accessible to those who may (or may not) be proficient with R, while still giving front end users the flexibility to explore particular models that they are comfortable with.

In the future I plan to add support for the following features:

- Parallel processing for Monte Carlo methods
- More general multilevel modeling support

References

- Bock, R. D., & Aitkin, M. (1981). Marginal maximum likelihood estimation of item parameters: Application of an EM algorithm. *Psychometrika, 46*, 443-459.
- Cai, L. (2010). High-Dimensional exploratory item factor analysis by a Metropolis-Hastings Robbins-Monro algorithm. *Psychometrika, 75*, 33-57.
- Chalmers, R. P. (2012). mirt: A Multidimensional Item Response Theory Package for the R Environment. *Journal of Statistical Software, 48*, 1-29.
- Fischer, G. H. (1983). Logistic latent trait models with linear constraints. *Psychometrika, 48*, 3-26.
- Gibbons, R. D., Darrell, R. B., Hedeker, D., ... (2007). Full-Information item bifactor analysis of graded response data. *Applied Psychological Measurement, 31*, 4-19