Item Response Theory in R: Estimation

Dr. Matthew Zeigenfuse

Lehrstuhl für Psychologisches Methodenlehre, Evaluation und Statistik
Psychologisches Institut
Universität Zürich

January 19, 2016

Overview

- Estimating the ability and item parameters in `mirt` is a two-step process

  Step 1 Estimate the item parameters using marginal maximum likelihood (MML) estimation via the `mirt` function

  Step 2 Estimate the ability parameters given the item parameters using one of the available ability estimators with the `fscores` function

Step 1:
Item Parameter Estimation

Ability in the Population

- The `mirt` package uses MML to estimate the item parameters of an IRT
- MML assumes that each test taker's ability is drawn from a larger population whose distribution is known
- The population distribution determines how many individuals of a given ability there are in the population
- For example, a population distribution might tell us that there are two times as many test takers of ability zero in the population than test takers of ability two

Standard Normal Population Distribution

- In practice, the population distribution of test taker ability is assumed to be a standard normal distribution
- A standard normal distribution is a normal distribution whose mean is zero and whose standard deviation is one
- This means that values close to zero occur more often in the population than values far frome zero

The Marginal Likelihood

- Assuming a population distribution allows us to compute the probability of observing the data matrix $\mathbf{X}$ from a population with the assumed distribution
- This is called the marginal likelihood
- Its expression for a standard normally distributed population is

$$P(\mathbf{X} \mid \psi) \propto \prod_{p=1}^{P} \int_{-\infty}^{\infty} \prod_{i=1}^{I} P(\mathbf{X}_{p\cdot} \mid \tilde{\theta}) \cdot \exp\left(-\frac{\tilde{\theta}^2}{2}\right) d\tilde{\theta}$$

- ψ is a placeholder for the item parameters of the IRT model
- Intuitively, the marginal likelihood of $\mathbf{X}_{p\cdot}$ is its average probability over the population

Marginal Maximum Likelihood

- MML finds the value $\hat{\boldsymbol{\psi}}$ of the item parameters that maximizes the marginal likelihood
- The standard error of the estimate is computed by inverting the negative of the Hessian matrix evaluated at $\hat{\boldsymbol{\psi}}$
- This value is a good guess when
  - Our IRT model is a reasonable approximation of reality
  - The population distribution is approximately correct

Computational Note

- The integral in the marginal likelihood must be computed numerically
- The `mirt` package provides a number of methods for computing this integral
- The default method (`"EM"`) is Bock & Aitkin's (1981) EM algorithm
- This method is sufficient when we are interested in estimating a single ability for the test takers

MML in R

- The `mirt` function performs MML estimation in the `mirt` package
- This function is very flexible, but at the very least you should provide the following arguments
  - `data`: The test responses to which you would like to fit an IRT model
  - `model`: Will always be "1" for unidimensional IRT models
  - `itemtype`: Character vector indicating the type (e.g., Rasch, PCM) of each of the test items.
- You can use the `guess` argument to fix the value of the pseudoguessing parameter in the 3PL

The `data` Argument

- The `data` argument contains the test responses
- It is either a matrix or data.frame with the structure described earlier:
  - Each row contains all of the data for one test taker
  - Each column contains all of the data for one test item
- Thus, `nrow(data)` is the number of test takers and `ncol(data)` is the number of test items
- Missing data should be coded as `NA`

The `itemtype` Argument

- The `itemtype` argument specifies each test item's type
- The item types covered today are `'Rasch'`, `'2PL'`, `'3PL'`, `'graded'`, `'grsm'`, `'gpcm'` and `'nominal'`
- The PCM can be obtained by setting `itemtype = 'Rasch'` with polytomous data
- The `itemtype` argument can be supplied in two ways
  - Using a character vector of length `ncol(data)` whose elements specify possible item types
  - Using a character vector of length 1 specifying an item type to be recycled for all items

Examples

- Fit the Rasch model for the `LSAT6` data set

```
> lsat6Full <- expand.table(LSAT6)
> lsat6Rasch <- mirt(lsat6Full, 1, "Rasch")
```

- Fit the graded response model to the `Science` data set

```
> sciGraded <- mirt(Science, 1, "graded")
```

Computing Standard Errors

- By default the `mirt` function does not compute standard errors, because it can be very time-consuming
- To compute standard errors, set `SE = TRUE`
- For example,

```
> lsat6Rasch <- mirt(lsat6Full, 1,
+                     "2PL", SE = TRUE)
```

Extracting Item Parameter Estimates (1)

- Item parameter estimates can be extracted from a model fit using the `coef` function
- Return a list with `ncol(data) + 1` in slope-intercept form. The first `ncol(data)` elements contain the item parameter estimates. The last element contains estimates of the normal population parameters

  ```
  > coef(lsat6Rasch)
  ```

- Return the estimates in standard form

  ```
  > coef(lsat6Rasch, IRTpars = TRUE)
  ```

- When all items have the same type, you can simplify the output by setting `simplify = TRUE`
- This returns a list with three elements

Extracting Item Parameter Estimates (2)

- When all items have the same type, you can set `simplify = TRUE`, e.g.,

  ```
  > ests <- coef(lsat6Rasch, IRTpars = TRUE,
  +                simplify = TRUE)
  > ests$items
  ```

  ```
         a      b g u
  Item_1 1 -2.731 0 1
  Item_2 1 -0.999 0 1
  Item_3 1 -0.240 0 1
  Item_4 1 -1.307 0 1
  Item_5 1 -2.100 0 1
  ```

- The remaining list elements are `means` and `cov` containing the mean and variance of the ability population distribution

Extracting Standard Errors

- The `coef` function can also be used to extract the standard errors when they were computed
- This can be done by setting `printSE = TRUE`
- For example,

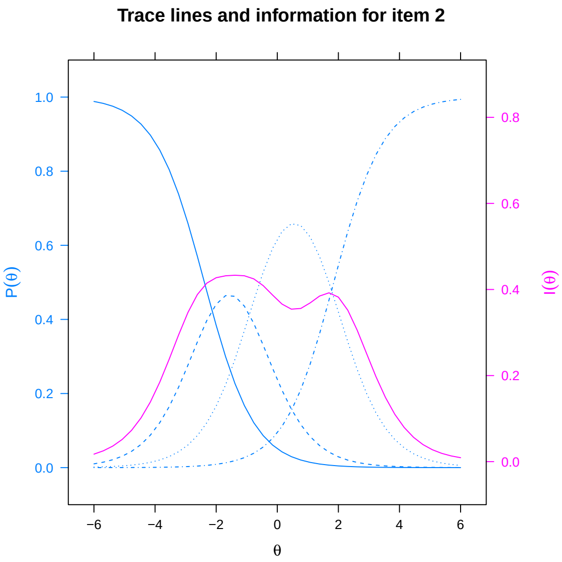
  `> coef(lsat6Rasch, IRTpars = TRUE, printSE = FALSE)`
- Note the `printSE = TRUE` will be ignored if `simplify = TRUE`

Visualizing Item Fits

- We can visualize aspects of the fits for each item using the `itemplot` function
- For unidimensional models, there are three primary arguments
  - `object`: An object returned by the `mirt` function
  - `item`: The item you would like to visualize
  - `type`: The type of plot you would like to see
    - `type = 'trace'` plots the ICC or category probability curves for the item
    - `type = 'info'` plots the item information
    - `type = 'infotrace'` plots both in a single window
- For example,

```
> itemplot(sciGraded, 2, "infotrace")
```

Example Output



Fixing Guessing or Upper Asymptote Parameters

- The `mirt` function allows for fixed guessing and upper asymptote parameters through its `guess` and `upper` arguments
- These parameters allow the user to use fixed, non-zero values for these parameters in conjunction with the 2PL model
- For example, the following fits the 3PL model with a fixed guessing parameter of 0.1 for all items

  ```
  > fit3plFix <- mirt(X, 1, "2PL", guess=0.1)
  ```
- Different guessing (upper) parameters can be supplied to different items by providing a vector of length `ncol(X)` to `guess` (`upper`)

Providing Item Blocks to GRSM

- The GRSM uses common threshold spacings for all items sharing the same rating scale
- These items are specified using the `grsm.block` argument
- This argument whose length equals the number of columns of the data matrix
- Items that are not contained within a GRSM block should be given the value `NA`
- Otherwise, blocks are labeled using integers: all items having the same integer code belong to the same block
- For example, setting `grsm.block` to `c(rep(1,3), rep(2,3), NA)` will create two blocks of three items and neither block will contain the 7th item

Constraints

-

Step 2:
Ability Estimation

Methods for Person Parameter Estimation

- The `mirt` package provides a number of methods for estimating the person parameter in the `fscores` function
  - Maximum likelihood (ML)
  - Maximum aposteriori (MAP)
  - Expected aposteriori (EAP), the default
  - Warm's (1989) weighted likelihood estimator (WLE)
- By default, the `fscores` function returns only the ability estimates as a matrix with P rows and 1 column
- To return the standard errors, set `full.scores.SE = TRUE`

Maximum Likelihood Estimation

- The ML estimate of the ability of test taker p is the value $\hat{\theta}_p$ maximizing the likelihood

$$P(\mathbf{X}_{p\cdot} \mid \theta_p) = \prod_{i=1}^{I} P(X_{pi} \mid \theta_p)$$

  for the previously estimated value of the item parameters
- Though simple, ML estimates have two problems:
  - They are not defined for test takers who correctly answer all items or incorrectly answer all items
  - They are relatively less accurate than the other estimators for short tests
- The following estimates the item parameters using ML for the LSAT6 data

```
> lsat6AbilML <- fscores(lsat6Rasch, method = "ML")
```

Bayesian Estimators (1)

- MAP and EAP are Bayesian estimators, so they are based on

$$P(\theta_p \mid \mathbf{X}_{p\cdot}) \propto P(\mathbf{X}_{p\cdot} \mid \theta_p)P(\theta_p)$$

- The quantity $P(\theta_p)$ is known as the prior distribution and is often taken to be the population distribution assumed in the previous section
- The quantity $P(\theta_p \mid \mathbf{X}_{p\cdot})$ is known as the posterior distribution
- The MAP estimator returns the value $\hat{\theta}_p$ maximizing the posterior distribution
- The EAP estimator returns the expect value (or average) of the posterior distribution

Bayesian Estimators (2)

- In practice, there should be little difference between the MAP and EAP for long tests
- For shorter tests, the EAP estimator will give more conservative estimates, because it takes the skew of the posterior into account
- The following estimates the item parameters using MAP for the LSAT6 data

  ```
  > lsat6AbilMAP <- fscores(lsat6Rasch, method = "MAP")
  ```

- The following estimates the item parameters using EAP for the LSAT6 data

  ```
  > lsat6AbilEAP <- fscores(lsat6Rasch, method = "EAP")
  ```

Weighted Likelihood Estimator

- Like the MAP estimator, Warm's WLE maximizes a weighted likelihood
$$P(X_{p\cdot} \mid \theta_p)h(\theta_p)$$
- The function $h(\theta_p)$ is chosen to produce more accurate estimates than the MLE for small sample sizes
- The following estimates the item parameters using WLE for the LSAT6 data

```
> lsat6AbilWLE <- fscores(lsat6Rasch, method = "WLE")
```

Summarizing and Visualizing Ability Estimates

- Ability estimates can be plotted and summarized using standard methods for vectors and matrices
- Some useful summary functions are `mean`, `sd`, `median`, `quantile` and `summary`
- For example, we can compute the 5th and 95th quantiles as follows

  ```
  > quantile(lsat6AbilWLE, c(.05, .95))
  ```
- We visualize the distribution of the abilities using a histogram:

  ```
  > hist(lsat6AbilWLE, xlab = "Ability")
  ```
- This is useful for checking for multimodality, which could like to poor estimates