Package ‘GGUM’

October 12, 2022

Title Generalized Graded Unfolding Model

Version 0.4-3

Description An implementation of the generalized graded unfolding model (GGUM) in R, see Roberts, Donoghue, and Laughlin (2000) <doi:10.1177/01466216000241001>). It allows to simulate data sets based on the GGUM. It fits the GGUM and the GUM, and it retrieves item and person parameter estimates. Several plotting functions are available (item and test information functions; item and test characteristic curves; item category response curves). Additionally, there are some functions that facilitate the communication between R and 'GGUM2004'. Finally, a model-fit checking utility, MODFIT(), is also available.

Imports stats, utils, psych, abind, viridis, Rdpack, xlsx

RdMacros Rdpack

Encoding UTF-8

License GPL (>= 2)

RoxygenNote 7.1.2

URL https://github.com/jorgetendeiro/GGUM/

NeedsCompilation no

Author Jorge N. Tendeiro [aut, cre],
     Sebastian Castro-Alvarez [aut]

Maintainer Jorge N. Tendeiro <tendeiro@hiroshima-u.ac.jp>

Repository CRAN

Date/Publication 2021-10-07 14:30:07 UTC

R topics documented:

GGUM-package ................................................................. 2
export.GGUM2004 ............................................................ 5
GenData.GGUM ............................................................... 6
GGUM ................................................................. 8
GUM ................................................................. 11
MODFIT .............................................................. 13
GGUM-package

Description

The GGUM package fits the generalized graded response model (GGUM; Roberts et al., 1996, 2000). It is based on marginal maximum likelihood (Roberts et al., 2000) to estimate the item parameters and an estimated a posteriori (EAP) method to estimate the person parameters.

The GGUM is defined by

\[ P(Z_i = z|\theta_n) = \frac{f(z) + f(M - z)}{\sum_{w=0}^{C} [f(w) + f(M - w)]}, \]

\[ f(w) = \exp \left\{ \alpha_i \left[ w(\theta_n - \delta_i) - \sum_{k=0}^{w} \tau_{ik} \right] \right\}, \]

where:

- The subscripts \( i \) and \( n \) identify the item and person, respectively.
- \( z = 0, \ldots, C \) denotes the observed answer response.
- \( M = 2C + 1 \) is the number of subjective response options minus 1.
- \( \theta_n \) is the latent trait score for person \( n \).
- \( \alpha_i \) is the item slope (discrimination).
- \( \delta_i \) is the item location.
- \( \tau_{ik} (k = 1, \ldots, M) \) are the threshold parameters.

Parameter \( \tau_{i0} \) is arbitrarily constrained to zero and the threshold parameters are constrained to symmetry around zero, that is, \( \tau_{i(C+1)} = 0 \) and \( \tau_{iz} = -\tau_{i(M-z+1)} \) for \( z \neq 0 \).

This package produces comparable results to the ones based on the GGUM2004 program (Roberts et al., 2000; Roberts et al., 2006), for the GUM (Model 3 in GGUM2004) and the GGUM (Model...
For those accustomed to using GGUM2004, this package provides a useful set of functions that allow exporting data and code to GGUM2004, running GGUM2004, and retrieving the parameter estimates. Thus, if desired, one can run GGUM2004 and retrieve the results completely from within the R environment.

Versions:

• Version 0.3.1 (January 2018)

• Version 0.3.2 (July 2018)  
  Fixed a bug related to data preprocessing (removing response patterns with all-disagree answers). Many thanks to JB Duck-Mayr for offering a fix in GitHub, and also to Michael Hermann who independently spotted the same issue (for dichotomous data).

• Version 0.4 (January 2020)  
  Fixed two bugs (in Theta.EAP() and write.GGUM2004())

• Version 0.4-1 (May 2020)  
  Implemented two changes in GUM.R to adapt to R 4.1 (currently R-devel) and survive CRAN’s build checks.

• Version 0.4-2 (February 2021)  
  Updated affiliation.

• Version 0.4-3 (October 2021)  
  Implemented further changes to adapt to R 4.1, similar to what was done in Version 0.4-1.

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>GGUM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>0.4-3</td>
</tr>
<tr>
<td>Date</td>
<td>2021-10-07</td>
</tr>
<tr>
<td>License</td>
<td>GPL Version 2 or later</td>
</tr>
</tbody>
</table>

The GGUM package contains useful functions, summarized below:

• Fitting the GUM/GGUM:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GenData.GGUM</td>
<td>Generate data from the GUM/GGUM</td>
</tr>
<tr>
<td>probs.GGUM</td>
<td>Compute model probabilities for the GGUM</td>
</tr>
<tr>
<td>GUM</td>
<td>Fit the GUM</td>
</tr>
<tr>
<td>GGUM</td>
<td>Fit the GGUM</td>
</tr>
<tr>
<td>MODFIT</td>
<td>MODFIT for the GGUM</td>
</tr>
<tr>
<td>Theta.EAP</td>
<td>Estimate thetas and their SEs (GUM, GGUM)</td>
</tr>
</tbody>
</table>

• Plots:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
</table>
plotCRC  Plot item category response curves (CRCs)
plotICC  Plot item characteristic curves (ICCs)
plotIIF  Plot item information functions (IIFs)
plotTCC  Plot test characteristic curve (TCC)
plotTIF  Plot test information function (TIF)

- GGUM2004 interface:

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>export.GGUM2004</td>
<td>Exports data in GGUM2004 friendly format</td>
</tr>
<tr>
<td>write.GGUM2004</td>
<td>Writes a command file for GGUM2004</td>
</tr>
<tr>
<td>run.GGUM2004</td>
<td>Call GGUM2004 and import the estimated parameters into R</td>
</tr>
<tr>
<td>read.item.GGUM2004</td>
<td>Read GGUM2004 item estimates into R</td>
</tr>
<tr>
<td>read.person.GGUM2004</td>
<td>Read GGUM2004 person estimates into R</td>
</tr>
</tbody>
</table>

- Available methods for objects of class "GGUM":

  plot()
  print()
  summary()

Author(s)

Maintainer: Jorge N. Tendeiro <tendeiro@hiroshima-u.ac.jp>

Authors:

  • Sebastian Castro-Alvarez <secastroal@gmail.com>

References


See Also

Useful links:

  • https://github.com/jorgetendeiro/GGUM/


## Examples

```r
## Not run:
# Example 1 - Same value C across items:
# Generate data:
gen1 <- GenData.GGUM(2000, 10, 2, seed = 125)
# Fit the GGUM:
fit1 <- GGUM(gen1$data, 2)
theta1 <- Theta.EAP(fit1)
# Plot the test information function:
plotTIF(fit1, theta1)
# Check model fit:
MOD.res <- MODFIT(fit1)

# Example 2 - Different C across items:
# Generate data:
set.seed(1); C <- sample(3:5, 10, replace = TRUE)
I <- 10
gen2 <- GenData.GGUM(2000, I, C, seed = 125)
# Fit the GGUM:
fit2 <- GGUM(gen2$data, C)
theta2 <- Theta.EAP(fit2)
# Plot item information functions for items 1 and 3:
plotIIF(fit2, theta2, items = c(1, 3))

# Example 3 - Fit GGUM using GGUM2004:
# Assuming the installation directory is C:/GGUM2004, then do this:
# Export data to GGUM2004:
export.GGUM2004(gen2$data)
# Write command file:
write.GGUM2004(I, C)
# Run GGUM2004:
res.GGUM2004 <- run.GGUM2004()
## End(Not run)
```

---

### Description

**export.GGUM2004** exports the data from R to a text file according to the format required by 'GGUM2004'.

### Usage

```r
export.GGUM2004(data, data.file = "data", data.dir = tempdir())
```

### Arguments

- **data**
  - The R data matrix to be exported.
data.file A character string defining the name of the data file. No file extension is required.

data.dir A character string defining the path to the directory where the data file (possibly exported by function `export.GGUM2004`) is stored. By default a temporary directory is used, but the user can naturally specify a directory of her choice.

Details

This function exports the R matrix data in 'GGUM2004' (Roberts, Donoghue, & Laughlin, 2000; Roberts et al., 2006) friendly format. This data file is to be used together with a 'GGUM2004' command script (or using the GUI itself, of course). 'GGUM2004' may be executed directly or may be called from R, see `run.GGUM2004`.

By experience, we noticed that long directory paths (especially if spaces are included) make 'GGUM2004' fail to execute with error `file not found`. Therefore, a good advice is to choose `data.dir` wisely (short paths, no spaces).

Observe that this function is optimized for the Windows operating system because 'GGUM2004' is a Windows program.

Author(s)

Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

References


Examples

```r
# Generate data:
C <- c(3, 3, 3, 5, 5, 5)
I <- 6
gen <- GenData.GGUM(750, I, C, seed = 125)
# Export data to 'GGUM2004':
export.GGUM2004(gen$data)
```

---

GenData.GGUM

Generate data from the GUM/GGUM

Description

GenData.GGUM generates all model parameters (items and persons) and item scores.
Usage

GenData.GGUM(N, I, C, model = "GGUM", seed = 123)

Arguments

N   Number of persons (rows).
I   Number of items (columns).
C   C is the number of observable response categories minus 1 (i.e., the item scores will be in the set \{0, 1, ..., C\}). It should either be a vector of I elements or a scalar. In the latter, case it is assumed that C applies to all items.
model A string identifying the model. Possible values are "GUM" or "GGUM" (default).
seed An integer, allowing the user to control the generation process (for replication purposes).

Value

The function returns a list with five elements:

- alpha.gen: The discrimination parameters.
- delta.gen: The difficulty parameters.
- taus.gen: The threshold parameters.
- theta.gen: The person parameters.
- data: The (NxI) data matrix. The item scores are coded 0, 1, ..., C for an item with (C+1) observable response categories.

Details

The generalized graded unfolding model (GGUM; Roberts & Laughlin, 1996; Roberts et al., 2000) is given by

\[
P(Z_i = z|\theta_n) = \frac{f(z) + f(M - z)}{\sum_{w=0}^{C} [f(w) + f(M - w)]},
\]

\[
f(w) = \exp\left\{\alpha_i \left[w(\theta_n - \delta_i) - \sum_{k=0}^{w} \tau_{ik}\right]\right\},
\]

where:

- The subscripts \(i\) and \(n\) identify the item and person, respectively.
- \(z = 0, \ldots, C\) denotes the observed answer response.
- \(M = 2C + 1\) is the number of subjective response options minus 1.
- \(\theta_n\) is the latent trait score for person \(n\).
- \(\alpha_i\) is the item slope (discrimination).
- \(\delta_i\) is the item location.
• \( \tau_{ik} \) \((k = 1, \ldots, M)\) are the threshold parameters.

Parameter \( \tau_{i0} \) is arbitrarily constrained to zero and the threshold parameters are constrained to symmetry around zero, that is, \( \tau_{i(C+1)} \) = 0 and \( \tau_{iz} = -\tau_{i(M-z+1)} \) for \( z \neq 0 \).

Parameters \( \alpha_i \) are randomly uniformly drawn from the (.5, 2) interval. Parameters \( \delta_i \) are randomly drawn from the standard normal distribution bounded between \(-2\) and \(2\). The threshold parameters are generated following the same procedure of Roberts, Donoghue, and Laughlin (2002). Finally, the person parameters are randomly drawn from the standard normal distribution.

If \( \text{model} = "\text{GUM}" \) the data based on the GUM (Roberts and Laughlin, 1996) model are generated. The GUM is a constrained version of the GGUM, where all discrimination parameters are equal to 1 and the item thresholds are shared by all items.

Author(s)

Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

Examples

```r
gen1 <- GenData.GGUM(500, 10, 5, seed = 456)
gen1$data # Retrieve the data.
gen1$alpha.gen # The discrimination parameters.

# Generate data based on items varying in the number of observable response categories:
gen2 <- GenData.GGUM(500, 5, c(5, 5, 5, 4, 4), seed = 789)
```

---

**GGUM**

*Fit the generalized graded unfolding model (GGUM)*

**Description**

GGUM estimates all item parameters for the GGUM.

**Usage**

```r
GGUM(
data,
C,
SE = TRUE,
precision = 4,
N.nodes = 30,
max.outer = 60,
max.inner = 60,
tol = 0.001
)
```
Arguments

data The $N \times I$ data matrix. The item scores are coded $0, 1, \ldots, C$ for an item with $(C + 1)$ observable response categories.

$C$ $C$ is the number of observable response categories minus 1 (i.e., the item scores will be in the set $\{0, 1, \ldots, C\}$). It should either be a vector of $I$ elements or a scalar. In the latter case, it is assumed that $C$ applies to all items.

SE Logical value: Estimate the standard errors of the item parameter estimates? Default is TRUE.

precision Number of decimal places of the results (default = 4).

N.nodes Number of nodes for numerical integration (default = 30).

max.outer Maximum number of outer iterations (default = 60).

max.inner Maximum number of inner iterations (default = 60).

tol Convergence tolerance (default = .001).

Value

The function returns a list (an object of class GGUM) with 12 elements:

data Data matrix.

C Vector $C$.

alpha The estimated discrimination parameters for the GGUM.

delta The estimated difficulty parameters.

taus The estimated threshold parameters.

SE The standard errors of the item parameters estimates.

rows.rm Indices of rows removed from the data before fitting the model, due to complete disagreement.

N.nodes Number of nodes for numerical integration.

tol.conv Loss function value at convergence (it is smaller than tol upon convergence).

iter.inner Number of inner iterations (it is equal to 1 upon convergence).

model Model fitted.

InformationCrit Loglikelihood, number of model parameters, AIC, BIC, CAIC.

Details

The generalized graded unfolding model (GGUM; Roberts & Laughlin, 1996; Roberts et al., 2000) is given by

$$P(Z_i = z|\theta_n) = \frac{f(z) + f(M - z)}{\sum_{w=0}^{C} [f(w) + f(M - w)]},$$

$$f(w) = exp \left\{ \alpha_i \left[ w(\theta_n - \delta_i) - \sum_{k=0}^{w} \tau_{ik} \right] \right\},$$

where:
The subscripts $i$ and $n$ identify the item and person, respectively.

- $z = 0, \ldots, C$ denotes the observed answer response.
- $M = 2C + 1$ is the number of subjective response options minus 1.
- $\theta_n$ is the latent trait score for person $n$.
- $\alpha_i$ is the item slope (discrimination).
- $\delta_i$ is the item location.
- $\tau_{ik}$ ($k = 1, \ldots, M$) are the threshold parameters.

Parameter $\tau_{i0}$ is arbitrarily constrained to zero and the threshold parameters are constrained to symmetry around zero, that is, $\tau_{i(C+1)} = 0$ and $\tau_{iz} = -\tau_{i(M-z+1)}$ for $z \neq 0$.

The marginal maximum likelihood algorithm of Roberts et al. (2000) was implemented.

**Author(s)**

Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

**References**


**Examples**

```r
## Not run:
# Example 1 - Same value C across items:
# Generate data:
gen1 <- GenData.GGUM(2000, 10, 2, seed = 125)
# Fit the GGUM:
fit1 <- GGUM(gen1$data, 2)
# Compare true and estimated item parameters:
cbind(gen1$alpha, fit1$alpha)
cbind(gen1$delta, fit1$delta)
cbind(c(gen1$taus[, 4:5]), c(fit1$taus[, 4:5]))

# Example 2 - Different C across items:
# Generate data:
set.seed(1); C <- sample(3:5, 10, replace = TRUE)
gen2 <- GenData.GGUM(2000, 10, C, seed = 125)
# Fit the GGUM:
fit2 <- GGUM(gen2$data, C)
# Compare true and estimated item parameters:
cbind(gen2$alpha, fit2$alpha)
cbind(gen2$delta, fit2$delta)
cbind(c(gen2$taus[, 7:11]), c(fit2$taus[, 7:11]))

## End(Not run)
```
**GUM**

*Fit the graded unfolding model (GUM)*

---

**Description**

`GUM` estimates all item parameters for the GUM.

**Usage**

```r
GUM(
  data,
  C,
  SE = TRUE,
  precision = 4,
  N.nodes = 30,
  max.outer = 60,
  max.inner = 60,
  tol = 0.001
)
```

**Arguments**

- **data**
  The $N \times I$ data matrix. The item scores are coded $0, 1, \ldots, C$ for an item with $(C + 1)$ observable response categories.

- **C**
  $C$ is the number of observable response categories minus 1 (i.e., the item scores will be in the set $\{0, 1, \ldots, C\}$). It should be a scalar since the GUM expects all items to be based on the same number of observable response categories.

- **SE**
  Logical value: Estimate the standard errors of the item parameter estimates? Default is `TRUE`.

- **precision**
  Number of decimal places of the results (default = 4).

- **N.nodes**
  Number of nodes for numerical integration (default = 30).

- **max.outer**
  Maximum number of outer iterations (default = 60).

- **max.inner**
  Maximum number of inner iterations (default = 60).

- **tol**
  Convergence tolerance (default = .001).

**Value**

The function returns a list (an object of class `GUM`) with 12 elements:

- **data**
  Data matrix.

- **C**
  Vector $C$.

- **alpha**
  In case of the GUM this is simply a vector of 1s.

- **delta**
  The estimated difficulty parameters.

- **taus**
  The estimated threshold parameters.
SE 
rows.rm
indices of rows removed from the data before fitting the model, due to complete disagreement.
N.nodes
number of nodes for numerical integration.
tol.conv
loss function value at convergence (it is smaller than tol upon convergence).
iter.inner
number of inner iterations (it is equal to 1 upon convergence).
model
model fitted.
InformationCrit
loglikelihood, number of model parameters, AIC, BIC, CAIC.

Details
The graded unfolding model (GUM; Roberts & Laughlin, 1996) is a constrained version of the GGUM (Roberts et al., 2000; see GGUM). GUM is constrained in two ways: All discrimination parameters are fixed to unity and the threshold parameters are shared across items. In particular, the last constraint implies that only data with the same response categories across items should be used (i.e., C is constant for all items).

Estimated GUM parameters are used as the second step of fitting the more general GGUM. Since under the GGUM data may include items with different number of response categories, the code to fitting the GUM was internally extended to accommodate for this.

The marginal maximum likelihood algorithm of Roberts et al. (2000) was implemented.

Author(s)
Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

References


Examples
# Generate data:
gen <- GenData.GGUM(400, 5, 3, "GUM", seed = 139)
# Fit the GUM:
fit <- GUM(gen$data, 3)
# Compare true and estimated item parameters:
cbind(gen$delta, fit$delta)
cbind(c(gen$taus[, 5:7]), c(fit$taus[, 5:7]))
MODFIT for the GGUM

Description

MODFIT computes the adjusted $\chi^2$ degrees of freedom ratios ($\chi^2/df$) introduced by Drasgow et al. (1995) for the GGUM.

Usage

MODFIT(IP, precision = 4)

Arguments

- **IP**: Object of class GGUM.
- **precision**: Number of decimal places of the results (default = 4).

Value

A list (an object of class MODFIT) with four elements: The results for singlets, doublets, triples, and a summary result.

Details

This function computes the adjusted $\chi^2$ degrees of freedom ratios ($\chi^2/df$) introduced by Drasgow et al. (1995). These $\chi^2$ statistics are based on expected frequencies that depend on the estimated item parameters and the distribution of $\theta$. The unadjusted statistic for item $i$ is given by

$$\chi^2_i = \sum_{z=0}^{C} \frac{(O_{iz} - E_{iz})^2}{E_{iz}},$$

where

- $O_{iz}$ is the observed frequency of choosing answer $z$ for item $i$ and $P_i(\theta)\varphi(\theta)d\theta$ is the standard normal density.
- The equation above applies to single items ('singlets'). The formula is easily extendible to pairs and triples of items. For a large number of items, the function selects suitable subsets of doublets and triples to perform the computations since its total number increases quickly with test length (Drasgow et al., 1995).

The formula is adjusted to a sample size of 3,000, as follows (see also LaHuis et al., 2011):

$$\frac{\chi^2_i}{df} = \frac{3,000(\chi^2_i - df)}{N} + df,$$
where \( df \) is a number of degrees of freedom that depends on the number of singlets, doublets, and triplets.

As an heuristic, values of \( \chi^2/df \) larger than 3 are indicative of model misfit.

This function produces the same numerical results as the MODFIT program (Stark, 2001) for the GGUM.

**Author(s)**

Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

**References**


**Examples**

```r
# For GUM:
# Generate data:
gen1 <- GenData.GGUM(400, 5, 3, "GUM", seed = 139)
# Fit the GUM:
fit1 <- GUM(gen1$data, 3)
# Compute the adjusted chi square degrees of freedom ratios:
modfit.res1 <- MODFIT(fit1)
modfit.res1$Singlets
modfit.res1$Doublets
modfit.res1$Triplets
modfit.res1$Summary

## Not run:
# For GGUM:
# Generate data:
set.seed(1); C <- sample(3:5, 10, replace = TRUE)
gen2 <- GenData.GGUM(2000, 10, C, "GGUM", seed = 156)
# Fit the GGUM:
fit2 <- GUM(gen2$data, C)
# Compute the adjusted chi square degrees of freedom ratios:
modfit.res2 <- MODFIT(fit1)
modfit.res2$Singlets
modfit.res2$Doublets
modfit.res2$Triplets
modfit.res2$Summary

## End(Not run)
```
**plotCRC**

*Plot item category response curves (CRCs)*

**Description**

`plotCRC` plots item CRCs for the GUM and the GGUM.

**Usage**

`plotCRC(IP, items = NULL, x.lim = 4, ThetaminDelta = TRUE, quiet = FALSE)`

**Arguments**

- **IP**: Object of class `GGUM`.
- **items**: Vector indicating the items for which the CRCs are to be plotted. Default is all items.
- **x.lim**: Controls the limits of the x-axis. Default is -4 through +4.
- **ThetaminDelta**: Logical; if `TRUE`, plot the CRCs centered at 0, otherwise plot the CRCs centered at $\delta$ (item's difficulty). Default is `TRUE`.
- **quiet**: Render all plots for `items` at once? Default is `FALSE`.

**Value**

The function returns a three-dimensional array with the probabilities associated to each item’s CRC. These are the values shown in the plot.

**Details**

This function plots the item category response curves (CRCs) for the requested items.

**Author(s)**

Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

**Examples**

```r
# For GUM:
# Generate data:
gen1 <- GenData.GGUM(400, 5, 3, "GUM", seed = 139)
# Fit the GUM:
fit1 <- GUM(gen1$data, 3)
# Plot CRCs:
plotCRC(fit1, items = 1, quiet = TRUE)
```

```r
# Not run:
# For GGUM:
# Generate data:
set.seed(1); C <- sample(3:5, 10, replace = TRUE)
```
```r
gen2 <- GenData.GGUM(2000, 10, C, "GGUM", seed = 156)
# Fit the GUM:
fit2 <- GGUM(gen2$data, C)
# Plot CRCs:
plotCRC(fit2, items = 1, quiet = TRUE)
```

---

**plotICC**

*Plot item characteristic curves (ICCs)*

**Description**

`plotICC` plots the ICCs for the GUM and the GGUM.

**Usage**

```r
plotICC(IP, Th, items = NULL, quiet = FALSE)
```

**Arguments**

- **IP**
  - Object of class `GGUM`.
- **Th**
  - Theta estimates from function `Theta.EAP()`.
- **items**
  - Vector indicating the items for which the ICCs are to be plotted. Default is all items.
- **quiet**
  - Render all plots for `items` at once? Default is `FALSE`.

**Value**

The function returns the correlation between observed and expected item scores (missing values pairwise removed).

**Details**

This function plots the item characteristic curves (ICCs).

**Author(s)**

Jorge N. Tendeiro, `<tendeiro@hiroshima-u.ac.jp>`
Examples

```r
## Not run:
# For GUM:
# Generate data
# (toy example: Too few items (due to computation time constraints) for
# accurate estimation of person parameters; larger number of items is
# required in practice):
g1 <- GenData.GGUM(400, 5, 3, "GUM", seed = 139)
# Fit the GUM:
f1 <- GUM(g1$data, 3)
th1 <- Theta.EAP(fit1)
# Plot ICCs:
plotICC(fit1, th1, items = 1, quiet = TRUE)

## End(Not run)
## Not run:
# For GGUM:
# Generate data:
set.seed(1); C <- sample(3:5, 10, replace = TRUE)
g2 <- GenData.GGUM(2000, 10, C, "GGUM", seed = 156)
# Fit the GGUM:
f2 <- GGUM(g2$data, C)
th2 <- Theta.EAP(fit2)
# Plot ICCs:
plotICC(fit2, th2, items = 1, quiet = TRUE)

## End(Not run)
```

plotIIF

*Plot item information functions (IIFs)*

Description

`plotIIF()` plots the IIFs for the GUM and the GGUM.

Usage

```r
plotIIF(IP, Th, items = NULL, quiet = FALSE)
```

Arguments

- **IP**: Object of class `GGUM`.
- **Th**: Theta estimates from function `Theta.EAP()`.
- **items**: Vector indicating the items for which the ICCs are to be plotted. Default is all items.
- **quiet**: Render all plots for items at once? Default is `FALSE`.
plotTCC

Value

The function returns the (x, y) coordinates of the IIFs.

Details

This function plots the item information functions (IIFs).

Author(s)

Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

Examples

## Not run:
# For GUM:
# Generate data
# (toy example: Too few items (due to computation time constraints) for
# accurate estimation of person parameters; larger number of items is
# required in practice):
gen1 <- GenData.GGUM(400, 5, 3, "GUM", seed = 139)
# Fit the GUM:
fit1 <- GUM(gen1$data, 3)
th1 <- Theta.EAP(fit1)
# Plot IIFs:
plotIIF(fit1, th1, items = 1, quiet = TRUE)

## End(Not run)
## Not run:
# For GGUM:
# Generate data:
set.seed(1); C <- sample(3:5, 10, replace = TRUE)
gen2 <- GenData.GGUM(2000, 10, C, "GGUM", seed = 156)
# Fit the GGUM:
fit2 <- GGUM(gen2$data, C)
th2 <- Theta.EAP(fit2)
# Plot IIFs:
plotIIF(fit2, th2, items = 1, quiet = TRUE)

## End(Not run)

plotTCC

Plot test characteristic curve (TCC)

Description

plot.TCC plots the TCC for the GUM and the GGUM.
**plotTCC**

### Usage

```r
plotTCC(IP, Th)
```

### Arguments

- **IP**: Object of class GGUM.
- **Th**: Theta estimates from function `Theta.EAP()`.

### Value

The function returns a list with three elements:

- **coords**: (x, y) coordinates of the TCC.
- **cor.OBS.EXP**: Correlation between observed and expected test scores (missing values pairwise removed).
- **cor.OBS.EXP.means**: Correlation between observed and expected mean test scores (missing values pairwise removed). The \( \theta \) interval between \(-4\) through \(+4\) is divided in 100 subintervals of equal length. The observed and expected mean scores are computed for each subinterval.

### Details

This function plots the test characteristic curve (TCC).

### Author(s)

Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

### Examples

```r
# Not run:
# For GUM:
# Generate data
# (toy example: Too few items (due to computation time constraints) for
# accurate estimation of person parameters; larger number of items is
# required in practice):
gen1 <- GenData.GGUM(400, 5, 3, "GUM", seed = 139)
# Fit the GUM:
fit1 <- GUM(gen1$data, 3)
th1 <- Theta.EAP(fit1)
# Plot TCC:
plotTCC(fit1, th1)

# End(Not run)
# Not run:
# For GGUM:
# Generate data:
set.seed(1); C <- sample(3:5, 10, replace = TRUE)
gen2 <- GenData.GGUM(2000, 10, C, "GGUM", seed = 156)
```
# Fit the GGUM:
fit2 <- GGUM(gen2$data, C)
th2 <- Theta.EAP(fit2)
# Plot TCC:
plotTCC(fit2, th2)

## End(Not run)

---

plotTIF

*Plot test information function (TIF)*

## Description

`plot.TIF` plots the TIF for the GUM and the GGUM.

## Usage

`plotTIF(IP, Th)`

## Arguments

- **IP**: Object of class `GGUM`.
- **Th**: Theta estimates from function `Theta.EAP()`.

## Value

The function returns the (x, y) coordinates of the TIF.

## Details

This function plots the test information function (TIF).

## Author(s)

Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

## Examples

```r
## Not run:
# For GUM:
# Generate data
# (toy example: Too few items (due to computation time constraints) for
# accurate estimation of person parameters; larger number of items is
# required in practice):
gen1 <- GenData.GGUM(400, 5, 3, "GUM", seed = 139)
# Fit the GUM:
fit1 <- GUM(gen1$data, 3)
th1 <- Theta.EAP(fit1)
```
# Plot TIF:
plotTIF(fit1, th1)

## End(Not run)

## Not run:
# For GGUM:
# Generate data:
set.seed(1); C <- sample(3:5, 10, replace = TRUE)
gen2 <- GenData.GGUM(2000, 10, C, "GGUM", seed = 156)
# Fit the GGUM:
fit2 <- GGUM(gen2$data, C)
th2 <- Theta.EAP(fit2)
# Plot TIF:
plotTIF(fit2, th2)

## End(Not run)

---

**probs.GGUM**

*Compute model probabilities for the GGUM*

**Description**

probs.GGUM computes model probabilities for the GGUM (and the GUM) for given item and person parameters.

**Usage**

probs.GGUM(alpha, delta, taus, theta, C)

**Arguments**

- **alpha**: A vector of length \( I \) with the discrimination parameters.
- **delta**: A vector of length \( I \) with the difficulty parameters.
- **taus**: An \( I \times M \) matrix with the threshold parameters (\( M = 2 \times \max C + 1 \)).
- **theta**: A vector of length \( N \) with the person parameters.
- **C**: \( C \) is the number of observable response categories minus 1 (i.e., the item scores will be in the set \( \{0, 1, ..., C\} \)). It should either be a vector of \( I \) elements or a scalar. In the latter case, it is assumed that \( C \) applies to all items.

**Value**

The function returns an \( N \times I \times K \) array with the GGUM probabilities, with \( K = \max C + 1 \). To retrieve the GUM-based probabilities just constrain alpha to a unit vector of length \( I \) (i.e., \( \text{alpha} = \text{rep}(1, I) \)). In this case, make sure \( C \) is constant across items.
Details

This function computes the GGUM-based probabilities for all (person, item, response category) combinations. For the GGUM formula see the help for function GGUM (GGUM).

Author(s)

Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

Examples

```r
C <- c(3, 3, 5, 5)
gen <- GenData.GGUM(10, 5, C, seed = 456)
gen.alpha <- gen$alpha.gen
gen.delta <- gen$delta.gen
gen.taus <- gen$taus.gen
gen.theta <- gen$theta.gen

# Compute model probabilities for the parameters above:
Ps <- probs.GGUM(gen.alpha, gen.delta, gen.taus, gen.theta, C)

# In particular, the sum of the probabilities across all response options
# (i.e., the third dimension) should be 1 for all (person, item) combinations:
apply(Ps, 1:2, sum)
```

Description

read.item.GGUM2004 reads the output files from 'GGUM2004' with the item parameters. Both the item parameter estimates and their standard errors are imported into R.

Usage

```r
read.item.GGUM2004(temp.dir = "C:/GGUM2004/TEMPFILE", precision = 4)
```

Arguments

- `temp.dir` The directory where 'GGUM2004' saved the output. By default, it is "C:/GGUM2004/TEMPFILE".
- `precision` Number of decimal places of the results (default = 4).

Value

read.item.GGUM2004 returns a list containing the following components:

- `alpha` The estimated discrimination parameters (for GGUM).
- `delta` The estimated difficulty parameters.
- `taus` The estimated threshold parameters.
alphaSE  The standard errors for the estimated discrimination parameters (for GGUM).
deltaSE  The standard errors for the estimated difficulty parameters.
tausSE  The standard errors for the estimated threshold parameters (above zero; recall that the
threshold parameters are constrained to symmetry around zero, that is,
\( \tau_i(C+1) = 0 \) and \( \tau_{iz} = -\tau_{i(M-z+1)} \) for \( z \neq 0 \).

Details
Observe that this function is optimized for the Windows operating system because 'GGUM2004' is a Windows program.

Author(s)
Sebastian Castro-Alvarez, <secastroal@gmail.com>

References

Examples
```r
## Not run:
# If the 'GGUM2004' output files are in the default directory
# (C:/GGUM2004/TEMPFILE), then simply execute
read.item.GGUM2004()

## End(Not run)
```

---

**read.person.GGUM2004**  
*Read 'GGUM2004' person estimates into R*

**Description**

read.person.GGUM2004 reads the output files from 'GGUM2004' with the person parameters. Both the person parameter estimates and their standard errors are imported into R.

**Usage**

```r
read.person.GGUM2004(temp.dir = "C:/GGUM2004/TEMPFILE", precision = 4)
```

**Arguments**

- **temp.dir**  The directory where 'GGUM2004' saved the output. By default, it is "C:/GGUM2004/TEMPFILE".
- **precision**  Number of decimal places of the results (default = 4).
Value

An $N \times 3$ matrix is returned, where $N$ is the number of persons. The first column is the person ID, the second column has the person parameter estimates, and the last column has the standard errors.

Details

Observe that this function is optimized for the Windows operating system because 'GGUM2004' is a Windows program.

Author(s)

Sebastian Castro-Alvarez, <secastroal@gmail.com>

References


Examples

```r
## Not run:
# If the 'GGUM2004' output files are in the default directory
# (C:/GGUM2004/TEMPFILE), then simply execute
read.person.GGUM2004()
## End(Not run)
```

---

**run.GGUM2004**

*Call 'GGUM2004' and import the estimated parameters into R*

Description

`run.GGUM2004` executes a previously exported 'GGUM2004' command file (via function `write.GGUM2004`). It returns the execution time, the item parameter estimates, and the person parameter estimates.

Usage

```r
run.GGUM2004(
    cmd.file = "cmd",
    data.file = "data",
    datacmd.dir = tempdir(),
    prog.dir = "C:/GGUM2004",
    precision = 4
)
```
**Arguments**

- **cmd.file**: A character string defining the name of the command file. No file extension is required.
- **data.file**: A character string defining the name of the data file. No file extension is required.
- **datacmd.dir**: A character string defining the path to the directory where both the data file (identified by the `data.file` parameter and exported by function `export.GGUM2004`) and the command file (identified by the `cmd.file` parameter and exported by function `write.GGUM2004`) are saved.
- **prog.dir**: A character string defining the directory where 'GGUM2004' is installed (default: "C:/GGUM2004").
- **precision**: Number of decimal places of the results (default = 4).

**Value**

`run.GGUM2004` returns a list containing the following components:

- **time**: The 'GGUM2004' execution time.
- **alpha**: The estimated discrimination parameters (for GGUM).
- **delta**: The estimated difficulty parameters.
- **taus**: The estimated threshold parameters.
- **SE**: The standard errors for the estimated item parameters.
- **theta**: The estimated person parameters and their standard errors.

**Details**

Function `run.GGUM2004` runs internally both functions `read.item.GGUM2004` (to import the 'GGUM2004' item estimates into R) and `read.person.GGUM2004` (to import the 'GGUM2004' person estimates into R).

By experience, we noticed that long directory paths (especially if spaces are included) make 'GGUM2004' fail to execute with error `file not found`. Therefore, a good advice is to choose `datacmd.dir` and `prog.dir` wisely (short paths, no spaces).

Observe that this function is optimized for the Windows operating system because 'GGUM2004' is a Windows program.

**Author(s)**

Sebastian Castro-Alvarez, <secastroal@gmail.com>

**References**

Examples

```r
## Not run:
# Generate data:
C <- c(3, 3, 3, 5, 5, 5)
I <- 6
gen <- GenData.GGUM(750, I, C, seed = 125)
# Export data to 'GGUM2004':
export.GGUM2004(gen$data)
# Write command file:
write.GGUM2004(I, C, model = "GGUM")
# Run 'GGUM2004':
res.GGUM2004 <- run.GGUM2004()

## End(Not run)
```

---

### Theta.EAP

**Estimate thetas and their SEs (GUM, GGUM)**

**Description**

Theta.EAP estimates the person theta parameters via EAP.

**Usage**

```r
Theta.EAP(IP, SE = TRUE, precision = 4, N.nodes = 30)
```

**Arguments**

- **IP** Object of class GGUM. The GUM/ GGUM estimated item parameters via functions GUM() / GGUM(), respectively.
- **SE** Logical value: Estimate the standard errors of the theta estimates? Default is TRUE.
- **precision** Number of decimal places of the results (default = 4).
- **N.nodes** Number of nodes for numerical integration (default = 30).

**Value**

If SE = TRUE, the function returns an $N \times 2$ matrix with two columns (thetas, SEs), where $N$ is the number of rows in the data matrix (i.e., persons). If SE = FALSE, the function returns the theta estimates as a vector of length $N$.

**Details**

The EAP procedure used here is based on Roberts, Donoghue, and Laughlin (2000), namely Equation 25 for the $\theta$ estimates and Equation 26 for corresponding standard errors. The EAP estimate is the posterior mean of the $\theta$ distribution for the corresponding response pattern. The standard error is computed as an approximation to the standard deviation of the posterior distribution. See Roberts et al. (2000) for more details.
write.GGUM2004

Author(s)

Jorge N. Tendeiro, <tendeiro@hiroshima-u.ac.jp>

Examples

# For GUM:
# Generate data
# (toy example: Too few items (due to computation time constraints) for
# accurate estimation of person parameters; larger number of items is
# required in practice):
gen1 <- GenData.GGUM(400, 5, 3, "GUM", seed = 139)
# Fit the GUM:
fit1 <- GUM(gen1$data, 3)
# Estimate the theta parameters:
Theta.EAP(fit1)
## Not run:
# For GGUM:
# Generate data:
set.seed(1); C <- sample(3:5, 10, replace = TRUE)
gen2 <- GenData.GGUM(2000, 10, C, "GGUM", seed = 156)
# Fit the GGUM:
fit2 <- GGUM(gen2$data, C)
# Estimate the theta parameters:
Theta.EAP(fit2)
## End(Not run)

write.GGUM2004

Writes a command file for ‘GGUM2004’

Description

write.GGUM2004 creates a ‘GGUM2004’ command file according to the test characteristics. The file is saved in the ‘GGUM2004’ predefined installation folder.

Usage

write.GGUM2004(
  I,
  C,
  cutoff = 2,
  model = "GGUM",
  cmd.file = "cmd",
  data.file = "data",
  data.dir = tempdir()
)
Arguments

I  The number of items.
C  \( C \) is the number of observable response categories minus 1 (i.e., the item scores will be in the set \( \{0, 1, \ldots, C\} \)). It should either be a vector of \( I \) elements or a scalar. In the latter case it is assumed that \( C \) applies to all items.
cutoff  Either a number or a vector of \( I \) elements which defines the cutoff value. Default is 2.
model  A string identifying the model. Possible values are "GUM" or "GGUM" (default).
cmd.file  A character string defining the name to give to the command file. No file extension is required.
data.file  A character string defining the name of the data file. No file extension is required.
data.dir  A character string defining the path to the directory where the data file (possibly exported by function `export.GGUM2004`) is stored. This also determines the directory where the command file generated by this function will be exported to. By default a temporary directory is used, but the user can naturally specify a directory of her choice.

Value

A script file is saved in the directory where 'GGUM2004' is installed.

Details

This function prepares a 'GGUM2004' friendly command script, which may be used to run the 'GGUM2004' program (Roberts, Donoghue, & Laughlin, 2000; Roberts et al., 2006). 'GGUM2004' may be executed directly or may be called from R, see `run.GGUM2004`.

By default and for convenience, the command script is saved in the same directory where the data file resides (data.dir). By experience, we noticed that long directory paths (especially if spaces are included) make 'GGUM2004' fail to execute with error `file not found`. Therefore, a good advice is to choose data.dir wisely (short paths, no spaces).

Observe that this function is optimized for the Windows operating system because 'GGUM2004' is a Windows program.

Author(s)

Sebastian Castro-Alvarez. <secastroal@gmail.com>

References


Examples

I <- 6
C <- c(3, 3, 3, 5, 5)
write.GGUM2004(I, C, model = "GGUM")
Index

_PKGAGE (GGUM-package), 2

export.GGUM2004, 4, 5, 6, 25, 28

GenData.GGUM, 3, 6
GGUM, 3, 8, 12, 22
GGUM-package, 2
GUM, 3, 11

MODFIT, 3, 13

plotCRC, 4, 15
plotICC, 4, 16
plotIIF, 4, 17
plotTCC, 4, 18
plotTIF, 4, 20
probs.GGUM, 3, 21

read.item.GGUM2004, 4, 22, 25
read.person.GGUM2004, 4, 23, 25
run.GGUM2004, 4, 6, 24, 28

Theta.EAP, 3, 26

write.GGUM2004, 4, 24, 25, 27