Package ‘ifaTools’

October 13, 2022

Date 2022-01-19
Title Toolkit for Item Factor Analysis with ‘OpenMx’
Version 0.23
Description Tools, tutorials, and demos of Item Factor Analysis using ‘OpenMx’.
This software is described in Pritikin & Falk (2020) <doi:10.1177/0146621620929431>.
License AGPL (>= 3)
VignetteBuilder knitr
Imports methods
Suggests testthat, roxygen2, knitr (>= 1.8), gridExtra, plyr, xtable
Depends shiny (>= 0.10), ggplot2, reshape2, rpf (>= 0.48), OpenMx (>= 2.3.1), R (>= 2.14.0)
URL https://github.com/jpritikin/ifaTools
RoxygenNote 7.1.1
NeedsCompilation no
Author Joshua N. Pritikin [cre, aut]
Maintainer Joshua N. Pritikin <jpritikin@pobox.com>
Repository CRAN
Date/Publication 2022-01-19 22:12:42 UTC

R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>addExploratoryFactors</td>
<td>2</td>
</tr>
<tr>
<td>iccPlot</td>
<td>2</td>
</tr>
<tr>
<td>itemModelExplorer</td>
<td>3</td>
</tr>
<tr>
<td>itemResponseMap</td>
<td>3</td>
</tr>
<tr>
<td>modelBuilder</td>
<td>4</td>
</tr>
<tr>
<td>plotInformation</td>
<td>4</td>
</tr>
<tr>
<td>replicateModelBy</td>
<td>5</td>
</tr>
<tr>
<td>SitemPlot</td>
<td>5</td>
</tr>
<tr>
<td>uniquenessPrior</td>
<td>6</td>
</tr>
<tr>
<td>univariatePrior</td>
<td>7</td>
</tr>
</tbody>
</table>
addExploratoryFactors

Description

Adds exploratory factors to a single factor model

Usage

addExploratoryFactors(model, toAdd, ..., addUniquenessPrior = TRUE)

Arguments

- model: a single factor (possibly multigroup) model
- toAdd: the number of factors to add
- ...: Not used. Forces remaining arguments to be specified by name.
- addUniquenessPrior: whether to add a uniqueness prior to the model (default TRUE)

iccPlot

Description

WARNING: This function is under development. The API may change in a future release.

Usage

iccPlot(grp, itemName, ..., width = 3, dataBins = 11, basis = c(1), factor = 1)

Arguments

- grp: an IFA group
- itemName: name of item to plot
- ...: Not used. Forces remaining arguments to be specified by name.
- width: sets the x axis to [-width,width]
- dataBins: number of partitions for the latent scores
- basis: the basis vector in the latent space
- factor: the score to use (TODO: should be a function of the basis vector?)
itemModelExplorer

A Shiny app to experiment with item models

Description

A Shiny app to experiment with item models

Usage

itemModelExplorer()

Examples

## Not run:
itemModelExplorer()  # will launch a browser in RStudio

## End(Not run)

itemResponseMap

Create item response map table

Description

Categories are placed at the mean score of the examinees who picked that category.

Usage

itemResponseMap(grp, ..., factor = 1)

Arguments

- **grp**
  - an IFA group

- **...**
  - Not used. Forces remaining arguments to be specified by name.

- **factor**
  - which factor to plot (defaults to 1)

Value

A data.frame of the raw data backing the plot. Item outcomes without any observations are omitted.
plotInformation

Description

Plot item information in the latent distribution

Usage

plotInformation(grp, ..., width = 3, showTotal = FALSE, basis = c(1))

Arguments

grp      an IFA group
...     Not used. Forces remaining arguments to be specified by name.
width    the plot will span from -width to width
showTotal whether to plot the total item information
basis    the basis vector (for multidimensional items)
replicateModelBy

Replicate a model for each group of data

Description
The reference group is fixed to a zero mean and identity covariance matrix.

Usage
replicateModelBy(
  tmpl,
  fullData,
  mMat,
  covMat,
  ...,            
  splitCol = "population",
  refGroup = "general",
  split = TRUE,
  compressData = TRUE
)

Arguments
   tmpl      an OpenMx model
   fullData  the complete data including the column indicating group membership
   mMat      an MxMatrix for latent means
   covMat    an MxMatrix for latent covariance
   ...       Not used. Forces remaining arguments to be specified by name.
   splitCol  the name of the column used to indicate group membership
   refGroup  the name of the reference group
   split     whether to split the data (defaults to TRUE)
   compressData whether to apply compressDataFrame (defaults to TRUE)

SitemPlot
Plot expected and observed table from SitemFit

Description
Plot expected and observed table from SitemFit

Usage
SitemPlot(sout, itemName, ..., showSampleSize = TRUE)
Arguments

- `sout`: output from SitemFit
- `itemName`: name of item to plot
- `...`: Not used. Forces remaining arguments to be specified by name.
- `showSampleSize`: whether to show the sample size at the top of the plot

### Description

To prevent Heywood cases, Bock, Gibbons, & Muraki (1988) suggested a beta prior on the uniqueness (Equations 43-46). The analytic gradient and Hessian are included for quick optimization using Newton-Raphson.

### Usage

```r
uniquenessPrior(model, numFactors, strength = 0.1, name = "uniquenessPrior")
```

### Arguments

- `model`: an `mxModel`
- `numFactors`: the number of factors. All items are assumed to have the same number of factors.
- `strength`: the strength of the prior
- `name`: the name of the `mxModel` that is returned

### Details

To reproduce these derivatives in maxima for the case of 2 slopes (c and d), use the following code:

```maxima
f(c,d) := -p*log(1-(c^2 / (c^2+d^2+1) + (d^2 / (c^2+d^2+1))));
diff(f(c,d), d),radcan;
diff(diff(f(c,d), d),d),radcan;
The general pattern is given in Bock, Gibbons, & Muraki.
```

### Value

An `mxModel` that evaluates to the prior density in deviance units

### References

Examples

```r
numItems <- 6
spec <- list()
spec[1:numItems] <- list(rpf.drm(factors=2))
names(spec) <- paste0("i", 1:numItems)
item <- mxMatrix(name="item", free=TRUE,
                 values=mxSimplify2Array(lapply(spec, rpf.rparam)))
item$labels[1:2,] <- paste0("/quotesingle.Var","p",1:(numItems * 2))
data <- rpf.sample(100, spec, item$values) # use a larger sample size
m1 <- mxModel(model="m1", item,
               mxData(observed=data, type="raw"),
               mxExpectationBA81(spec),
               mxFitFunctionML())
up <- uniquenessPrior(m1, 2)
container <- mxModel("container", m1, up,
                    mxFitFunctionMultigroup(c("m1", "uniquenessPrior")),
                    mxComputeSequence(list(
                           mxComputeOnce("fitfunction", c('fit','gradient')),
                           mxComputeReportDeriv())))
container <- mxRun(container)
container$output$fit
container$output$gradient
```

---

**univariatePrior**

Univariate priors commonly used in IFA models

### Description

The returned model evaluates to the fit of the priors in deviance (-2 log likelihood) units. The analytic gradient and Hessian are included for quick optimization using Newton-Raphson.

### Usage

```r
univariatePrior(type, labels, mode, strength = NULL, name = "univariatePrior")
```

### Arguments

- **type**: one of c("lnorm","beta","logit-norm")
- **labels**: a vector of parameters to which to apply the prior density
- **mode**: the mode of the prior density
- **strength**: a prior-specific strength (optional)
- **name**: the name of the mxModel returned

### Details

Priors of type 'beta' and 'logit-norm' are commonly used for the lower asymptote parameter of the 3PL model. Both of these priors assume that the parameter is in logit units. The 'lnorm' prior can be used for slope parameters.
Value

an mxModel that evaluates to the prior density in deviance units

Examples

```r
model <- univariatePrior("logit-norm", "x1", -1)
model$priorParam$values[1,1] <- -.6
model <- mxRun(model)
model$output$fit
model$output$gradient
model$output$hessian
```
Index

addExploratoryFactors, 2
iccPlot, 2
itemModelExplorer, 3
itemResponseMap, 3
modelBuilder, 4
plotInformation, 4
replicateModelBy, 5
SitemPlot, 5
uniquenessPrior, 6
univariatePrior, 7