

# Package ‘rpf’

August 29, 2016

**Type** Package

**Title** Response Probability Functions

**Version** 0.53

**Date** 2016-06-28

**Maintainer** Joshua Pritikin <jpritikin@pobox.com>

**Description** The purpose of this package is to factor out logic and math common to Item Factor Analysis fitting, diagnostics, and analysis. It is envisioned as core support code suitable for more specialized IRT packages to build upon. Complete access to optimized C functions are made available with `R_RegisterCCallable()`.

**LazyData** yes

**LazyDataCompression** bzip2

**License** GPL (>= 3)

**LinkingTo** RcppEigen

**Suggests** testthat, roxygen2, ggplot2, reshape2, gridExtra, numDeriv,  
knitr, mirt

**Imports** mvtnorm

**Depends** methods, parallel, R (>= 2.14.0)

**VignetteBuilder** knitr

**Collate** 'init.R' 'classes.R' 'fit.R' 'drm.R' 'nrm.R' 'mcm.R' 'grm.R'  
'LSAT.R' 'sample.R' 'dataframe.R' 'diagnose.R' 'science.R'  
'kct.R' 'openmx.R' 'flexmirt.R' 'util.R' 'Imp.R'

**URL** <https://github.com/jpritikin/rpf>

**RoxygenNote** 5.0.1

**NeedsCompilation** yes

**Author** Joshua Pritikin [cre, aut],  
Jonathan Weeks [ctb],  
Li Cai [ctb],  
Carrie Houts [ctb],  
Phil Chalmers [ctb],

Michael D. Hunter [ctb],  
 Carl F. Falk [ctb]

**Repository** CRAN

**Date/Publication** 2016-06-28 08:45:48

## R topics documented:

An introduction . . . . .	3
as.IFAGroup . . . . .	4
bestToOmit . . . . .	5
ChenThissen1997 . . . . .	6
Class rpf.1dim . . . . .	7
Class rpf.1dim.drm . . . . .	7
Class rpf.1dim.graded . . . . .	7
Class rpf.1dim.grm . . . . .	7
Class rpf.1dim.lmp . . . . .	8
Class rpf.base . . . . .	8
Class rpf.mdim . . . . .	8
Class rpf.mdim.drm . . . . .	8
Class rpf.mdim.graded . . . . .	8
Class rpf.mdim.grm . . . . .	9
Class rpf.mdim.mcm . . . . .	9
Class rpf.mdim.nrm . . . . .	9
compressDataFrame . . . . .	9
crosstabTest . . . . .	10
EAPscores . . . . .	10
expandDataFrame . . . . .	11
fromFactorLoading . . . . .	12
fromFactorThreshold . . . . .	12
itemOutcomeBySumScore . . . . .	13
kct . . . . .	13
logit . . . . .	14
LSAT6 . . . . .	14
LSAT7 . . . . .	15
multinomialFit . . . . .	15
observedSumScore . . . . .	16
omitItems . . . . .	17
omitMostMissing . . . . .	17
orderCompletely . . . . .	18
ordinal.gamma . . . . .	18
ptw2011.gof.test . . . . .	19
read.flexmirt . . . . .	20
rpf.1dim.fit . . . . .	20
rpf.1dim.moment . . . . .	21
rpf.1dim.residual . . . . .	22
rpf.1dim.stdresidual . . . . .	22
rpf.dLL . . . . .	23

rpf.drm . . . . .	24
rpf.dTheta . . . . .	25
rpf.grm . . . . .	25
rpf.id_of . . . . .	26
rpf.info . . . . .	27
rpf.lmp . . . . .	27
rpf.logprob . . . . .	29
rpf.mcm . . . . .	30
rpf.mean.info . . . . .	31
rpf.mean.info1 . . . . .	31
rpf.modify . . . . .	32
rpf.nrm . . . . .	32
rpf.numParam . . . . .	33
rpf.numSpec . . . . .	34
rpf.ogive . . . . .	34
rpf.paramInfo . . . . .	35
rpf.prob . . . . .	36
rpf.rescale . . . . .	36
rpf.rparam . . . . .	37
rpf.sample . . . . .	38
science . . . . .	39
SitemFit . . . . .	39
SitemFit1 . . . . .	40
stripData . . . . .	41
sumScoreEAP . . . . .	41
sumScoreEAPTest . . . . .	42
tabulateRows . . . . .	43
toFactorLoading . . . . .	43
toFactorThreshold . . . . .	44
write.flexmirt . . . . .	45
<b>Index</b>	<b>46</b>

**Description**

The purpose of this package is to factor out logic and math common to Item Factor Analysis fitting, diagnostics, and analysis. It is envisioned as core support code suitable for more specialized IFA packages to build upon.

## Details

This package provides optimized, low-level functions to map parameters to response probabilities for dichotomous (1PL, 2PL and 3PL) `rpf.drm` and polytomous (graded response `rpf.grm`, partial credit/generalized partial credit (via the nominal model), and nominal `rpf.nrm` items.

Item model parameters are passed around as a numeric vector. A 1D matrix is also acceptable. Regardless of model, parameters are always ordered as follows: discrimination/slope ("a"), difficulty/intercept ("b"), and pseudo guessing/upper-bound ("g"/"u"). If person ability ranges from negative to positive then probabilities are output from incorrect to correct. That is, a low ability person (e.g., ability = -2) will be more likely to get an item incorrect than correct. For example, a dichotomous model that returns [.25, .75] indicates a probability of .25 for incorrect and .75 for correct. A polytomous model will have the most incorrect probability at index 1 and the most correct probability at the maximum index.

All models are always in the logistic metric. To obtain normal ogive discrimination parameters, divide slope parameters by `rpf.ogive`. Item models are estimated in slope-intercept form. Input/output matrices arranged in the way most convenient for low-level processing in C. The maximum absolute logit is 35 because  $f(x) := 1 - \exp(x)$  loses accuracy around  $f(-35)$  and equals 1 at  $f(-38)$  due to the limited accuracy of double precision floating point.

This package could also accrete functions to support plotting (but not the actual plot functions).

## References

Thissen, D. and Steinberg, L. (1986). A taxonomy of item response models. *Psychometrika* 51(4), 567-577.

## See Also

See `rpf.rparam` to create item parameters.

---

as.IFAGroup

*Convert an OpenMx MxModel object into an IFA group*

---

## Description

When “minItemsPerScore” is passed, EAP scores will be computed from the data and stored. Scores are required for some diagnostic tests. See discussion of “minItemsPerScore” in [EAPscores](#).

## Usage

```
as.IFAGroup(mxModel, data = NULL, container = NULL, ...,
  minItemsPerScore = NULL)
```

**Arguments**

mxModel	MxModel object
data	observed data (otherwise the data will be taken from the mxModel)
container	an MxModel in which to search for the latent distribution matrices
...	Not used. Forces remaining arguments to be specified by name.
minItemsPerScore	minimum number of items required to compute a score (also see description)

**Value**

a groups with item parameters and latent distribution

**See Also**

[ifaTools](#)

---

bestToOmit	<i>Identify the columns with most missing data</i>
------------	--

---

**Description**

If a reference column is given then only rows that are not missing on the reference column are considered. Otherwise all rows are considered.

**Usage**

```
bestToOmit(grp, omit, ref = NULL)
```

**Arguments**

grp	an IFA group
omit	the maximum number of items to omit
ref	the reference column (optional)

ChenThissen1997

*Computes local dependence indices for all pairs of items***Description**

Item Factor Analysis makes two assumptions: (1) that the latent distribution is reasonably approximated by the multivariate Normal and (2) that items are conditionally independent. This test examines the second assumption. The presence of locally dependent items can inflate the precision of estimates causing a test to seem more accurate than it really is.

**Usage**

```
ChenThissen1997(grp, ..., data = NULL, inames = NULL, qwidth = 6,
  qpoints = 49, method = "pearson", .twotier = TRUE, .parallel = TRUE)
```

**Arguments**

grp	a list with the spec, param, mean, and cov describing the group
...	Not used. Forces remaining arguments to be specified by name.
data	data
inames	a subset of items to examine
qwidth	quadrature width
qpoints	number of equally spaced quadrature points
method	method to use to calculate P values. The default is the Pearson $X^2$ statistic. Use "lr" for the similar likelihood ratio statistic.
.twotier	whether to enable the two-tier optimization
.parallel	whether to take advantage of multiple CPUs (default TRUE)

**Details**

Statically significant entries suggest that the item pair has local dependence. Since  $\log(.01)=-4.6$ , an absolute magnitude of 5 is a reasonable cut-off. Positive entries indicate that the two item residuals are more correlated than expected. These items may share an unaccounted for latent dimension. Consider a redesign of the items or the use of testlets for scoring. Negative entries indicate that the two item residuals are less correlated than expected.

**Value**

a list with raw, pval and detail. The pval matrix is a lower triangular matrix of log P values with the sign determined by relative association between the observed and expected tables (see [ordinal.gamma](#))

**References**

- Chen, W.-H. & Thissen, D. (1997). Local dependence indexes for item pairs using Item Response Theory. *Journal of Educational and Behavioral Statistics*, 22(3), 265-289.
- Thissen, D., Steinberg, L., & Mooney, J. A. (1989). Trace lines for testlets: A use of multiple-categorical-response models. *Journal of Educational Measurement*, 26 (3), 247–260.
- Wainer, H. & Kiely, G. L. (1987). Item clusters and computerized adaptive testing: A case for testlets. *Journal of Educational measurement*, 24(3), 185–201.

**See Also**

[ifaTools](#)

---

Class rpf.1dim      *The base class for 1 dimensional response probability functions.*

---

**Description**

The base class for 1 dimensional response probability functions.

---

Class rpf.1dim.drm      *Unidimensional dichotomous item models (1PL, 2PL, and 3PL).*

---

**Description**

Unidimensional dichotomous item models (1PL, 2PL, and 3PL).

---

Class rpf.1dim.graded      *The base class for 1 dimensional graded response probability functions.*

---

**Description**

This class contains methods common to both the generalized partial credit model and the graded response model.

---

Class rpf.1dim.grm      *The unidimensional graded response item model.*

---

**Description**

The unidimensional graded response item model.

---

Class `rpf.1dim.1mp`      *Unidimensional logistic function of a monotonic polynomial.*

---

### **Description**

Unidimensional logistic function of a monotonic polynomial.

---

Class `rpf.base`      *The base class for response probability functions.*

---

### **Description**

Item specifications should not be modified after creation.

---

Class `rpf.mdim`      *The base class for multi-dimensional response probability functions.*

---

### **Description**

The base class for multi-dimensional response probability functions.

---

Class `rpf.mdim.drm`      *Multidimensional dichotomous item models (M1PL, M2PL, and M3PL).*

---

### **Description**

Multidimensional dichotomous item models (M1PL, M2PL, and M3PL).

---

Class `rpf.mdim.graded`      *The base class for multi-dimensional graded response probability functions.*

---

### **Description**

This class contains methods common to both the generalized partial credit model and the graded response model.



---

Class `rpf.mdim.grm`     *The multidimensional graded response item model.*

---

**Description**

The multidimensional graded response item model.

---

Class `rpf.mdim.mcm`     *The multiple-choice response item model (both unidimensional and multidimensional models have the same parameterization).*

---

**Description**

The multiple-choice response item model (both unidimensional and multidimensional models have the same parameterization).

---

Class `rpf.mdim.nrm`     *The nominal response item model (both unidimensional and multidimensional models have the same parameterization).*

---

**Description**

The nominal response item model (both unidimensional and multidimensional models have the same parameterization).

---

`compressDataFrame`     *Compress a data frame into unique rows and frequencies*

---

**Description**

Compress a data frame into unique rows and frequency counts.

**Usage**

```
compressDataFrame(tabdata, freqColName = "freq")
```

**Arguments**

`tabdata`             An object of class `data.frame`  
`freqColName`         Column name to contain the frequencies

**Value**

Returns a compressed data frame

**Examples**

```
df <- as.data.frame(matrix(c(sample.int(2, 30, replace=TRUE)), 10, 3))
compressDataFrame(df)
```

---

crosstabTest	<i>Monte-Carlo test for cross-tabulation tables</i>
--------------	---

---

**Description**

This is for developers.

**Usage**

```
crosstabTest(ob, ex, trials)
```

**Arguments**

ob	observed table
ex	expected table
trials	number of Monte-Carlo trials

---

EAPscores	<i>Compute EAP scores</i>
-----------	---------------------------

---

**Description**

If you have missing data then you must specify `minItemsPerScore`. This option will set scores to NA when there are too few items to make an accurate score estimate. If you are using the scores as point estimates without considering the standard error then you should set `minItemsPerScore` as high as you can tolerate. This will increase the amount of missing data but scores will be more accurate. If you are carefully considering the standard errors of the scores then you can set `minItemsPerScore` to 1. This will mimic the behavior of most other IFA software wherein scores are estimated if there is at least 1 non-NA item for the score. However, it may make more sense to set `minItemsPerScore` to 0. When set to 0, all NA rows are scored to the prior distribution.

**Usage**

```
EAPscores(grp, ..., naAction = NULL, compressed = FALSE)
```

**Arguments**

grp	a list with spec, param, data, and minItemsPerScore
...	Not used. Forces remaining arguments to be specified by name.
naAction	deprecated, will be removed in the next release
compressed	output one score per observed data row even when weightColumn is set (default FALSE)

**Examples**

```
spec <- list()
spec[1:3] <- rpf.grm(outcomes=3)
param <- sapply(spec, rpf.rparam)
data <- rpf.sample(5, spec, param)
colnames(param) <- colnames(data)
grp <- list(spec=spec, param=param, data=data, minItemsPerScore=1L)
EAPscores(grp)
```

---

expandDataFrame	<i>Expand summary table of patterns and frequencies</i>
-----------------	---

---

**Description**

Expand a summary table of unique response patterns to a full sized data-set.

**Usage**

```
expandDataFrame(tabdata, freqName = NULL)
```

**Arguments**

tabdata	An object of class data.frame with the unique response patterns and the number of frequencies
freqName	Column name containing the frequencies

**Value**

Returns a data frame with all the response patterns

**Author(s)**

Based on code by Phil Chalmers <rphilip.chalmers@gmail.com>

**Examples**

```
data(LSAT7)
expandDataFrame(LSAT7, freqName="freq")
```

---

fromFactorLoading      *Convert factor loadings to response function slopes*

---

**Description**

Convert factor loadings to response function slopes

**Usage**

```
fromFactorLoading(loading, ogive = rpf.ogive)
```

**Arguments**

loading            a matrix with items in the rows and factors in the columns  
 ogive             the ogive constant (default rpf.ogive)

**Value**

a slope matrix with items in the columns and factors in the rows

---

fromFactorThreshold      *Convert factor thresholds to response function intercepts*

---

**Description**

Convert factor thresholds to response function intercepts

**Usage**

```
fromFactorThreshold(threshold, loading, ogive = rpf.ogive)
```

**Arguments**

threshold         a matrix with items in the columns and thresholds in the rows  
 loading           a matrix with items in the rows and factors in the columns  
 ogive             the ogive constant (default rpf.ogive)

**Value**

an item intercept matrix with items in the columns and intercepts in the rows

---

itemOutcomeBySumScore *Produce an item outcome by observed sum-score table*

---

### Description

Produce an item outcome by observed sum-score table

### Usage

```
itemOutcomeBySumScore(grp, mask, interest)
```

### Arguments

grp	a list with spec, param, and data
mask	a vector of logicals indicating which items to include
interest	index or name of the item of interest

### Examples

```
set.seed(1)
spec <- list()
spec[1:3] <- rpf.grm(outcomes=3)
param <- sapply(spec, rpf.rparam)
data <- rpf.sample(5, spec, param)
colnames(param) <- colnames(data)
grp <- list(spec=spec, param=param, data=data)
itemOutcomeBySumScore(grp, c(FALSE,TRUE,TRUE), 1L)
```

---

kct	<i>Knox Cube Test dataset</i>
-----	-------------------------------

---

### Description

These data from Wright & Stone (1979, p. 31) were fit with Winsteps 3.73 using a 1PL model (slope fixed to 1).

### References

Wright, B. D. & Stone, M. H. (1979). *Best Test Design: Rasch Measurement*. Univ of Chicago Social Research.

### Examples

```
data(kct)
```

---

logit	<i>Transform from [0,1] to the reals</i>
-------	--

---

**Description**

The logit function is a standard transformation from [0,1] (such as a probability) to the real number line. This function is exactly the same as qlogis.

**Usage**

```
logit(p, location = 0, scale = 1, lower.tail = TRUE, log.p = FALSE)
```

**Arguments**

p	a number between 0 and 1
location	see qlogis
scale	see qlogis
lower.tail	see qlogis
log.p	see qlogis

**See Also**

qlogis, plogis

**Examples**

```
logit(.5) # 0
logit(.25) # -1.098
logit(0) # -Inf
```

---

LSAT6	<i>Description of LSAT6 data</i>
-------	----------------------------------

---

**Description**

Data from Thissen (1982); contains 5 dichotomously scored items obtained from the Law School Admissions Test, section 6.

**Author(s)**

Phil Chalmers <rphilip.chalmers@gmail.com>

**References**

Thissen, D. (1982). Marginal maximum likelihood estimation for the one-parameter logistic model. *Psychometrika*, 47, 175-186.

**Examples**

```
data(LSAT6)
```

---

LSAT7

*Description of LSAT7 data***Description**

Data from Bock & Lieberman (1970); contains 5 dichotomously scored items obtained from the Law School Admissions Test, section 7.

**Author(s)**

Phil Chalmers <rphilip.chalmers@gmail.com>

**References**

Bock, R. D., & Lieberman, M. (1970). Fitting a response model for  $n$  dichotomously scored items. *Psychometrika*, 35(2), 179-197.

**Examples**

```
data(LSAT7)
```

---

multinomialFit

*Multinomial fit test***Description**

For degrees of freedom, we use the number of observed statistics (incorrect) instead of the number of possible response patterns (correct) (see Bock, Gibbons, & Muraki, 1998, p. 265). This is not a huge problem because this test becomes poorly calibrated when the multinomial table is sparse. For more accurate p-values, you can conduct a Monte-Carlo simulation study (see examples).

**Usage**

```
multinomialFit(grp, independenceGrp, ..., method = "lr", log = TRUE,
               .twotier = TRUE)
```

**Arguments**

grp	a list with the spec, param, mean, and cov describing the group
independenceGrp	a list with the spec, param, mean, and cov describing the independence group
...	Not used. Forces remaining arguments to be specified by name.
method	lr (default) or pearson
log	whether to report p-value in log units
.twotier	whether to use the two-tier optimization (default TRUE)

**Details**

Rows with missing data are ignored.

The full information test is described in Bartholomew & Tzamourani (1999, Section 3).

For CFI and TLI, you must provide an independence model.

**References**

Bartholomew, D. J., & Tzamourani, P. (1999). The goodness-of-fit of latent trait models in attitude measurement. *Sociological Methods and Research*, 27(4), 525-546.

Bock, R. D., Gibbons, R., & Muraki, E. (1988). Full-information item factor analysis. *Applied Psychological Measurement*, 12(3), 261-280.

**Examples**

```
# Create an example IFA group
grp <- list(spec=list())
grp$spec[1:10] <- rpf.grm()
grp$param <- sapply(grp$spec, rpf.rparam)
colnames(grp$param) <- paste("i", 1:10, sep="")
grp$mean <- 0
grp$cov <- diag(1)
grp$uniqueFree <- sum(grp$param != 0)
grp$data <- rpf.sample(1000, grp=grp)

# Monte-Carlo simulation study
mcReps <- 3 # increase this to 10,000 or so
stat <- rep(NA, mcReps)
for (rx in 1:mcReps) {
  t1 <- grp
  t1$data <- rpf.sample(grp=grp)
  stat[rx] <- multinomialFit(t1)$statistic
}
sum(multinomialFit(grp)$statistic > stat)/mcReps # better p-value
```

---

observedSumScore	<i>Compute the observed sum-score</i>
------------------	---------------------------------------

---

**Description**

Compute the observed sum-score

**Usage**

```
observedSumScore(grp, ..., mask, summary = TRUE)
```



**Arguments**

grp	a list with spec, param, and data
...	Not used. Forces remaining arguments to be specified by name.
mask	a vector of logicals indicating which items to include
summary	whether to return a summary (default) or per-row scores

**Examples**

```
spec <- list()
spec[1:3] <- rpf.grm(outcomes=3)
param <- sapply(spec, rpf.rparam)
data <- rpf.sample(5, spec, param)
colnames(param) <- colnames(data)
grp <- list(spec=spec, param=param, data=data)
observedSumScore(grp)
```

---

omitItems	<i>Omit the given items</i>
-----------	-----------------------------

---

**Description**

Omit the given items

**Usage**

```
omitItems(grp, excl)
```

**Arguments**

grp	an IFA group
excl	vector of column names to omit

---

omitMostMissing	<i>Omit items with the most missing data</i>
-----------------	--

---

**Description**

Items with no missing data are never omitted, regardless of the number of items requested.

**Usage**

```
omitMostMissing(grp, omit)
```

**Arguments**

grp	an IFA group
omit	the maximum number of items to omit

orderCompletely      *Order a data.frame by missingness and all columns*

---

**Description**

Completely order all rows in a data.frame.

**Usage**

```
orderCompletely(observed)
```

**Arguments**

observed      a data.frame holding ordered factors in every column

**Value**

the sorted order of the rows

**Examples**

```
df <- as.data.frame(matrix(c(sample.int(2, 30, replace=TRUE)), 10, 3))
mask <- matrix(c(sample.int(3, 30, replace=TRUE)), 10, 3) == 1
df[mask] <- NA
df[orderCompletely(df),]
```

---

ordinal.gamma      *Compute the ordinal gamma association statistic*

---

**Description**

Compute the ordinal gamma association statistic

**Usage**

```
ordinal.gamma(mat)
```

**Arguments**

mat      a cross tabulation matrix

**References**

Agresti, A. (1990). Categorical data analysis. New York: Wiley.

**Examples**

```
# Example data from Agresti (1990, p. 21)
jobsat <- matrix(c(20,22,13,7,24,38,28,18,80,104,81,54,82,125,113,92), nrow=4, ncol=4)
ordinal.gamma(jobsat)
```

---

ptw2011.gof.test	<i>Compute the P value that the observed and expected tables come from the same distribution</i>
------------------	--

---

**Description**

This test is an alternative to Pearson's  $X^2$  goodness-of-fit test. In contrast to Pearson's  $X^2$ , no ad hoc cell collapsing is needed to avoid an inflated false positive rate in situations of sparse cell frequencies. The statistic rapidly converges to the Monte-Carlo estimate as the number of draws increases.

**Usage**

```
ptw2011.gof.test(observed, expected)
```

**Arguments**

observed	observed matrix
expected	expected matrix

**Value**

The P value indicating whether the two tables come from the same distribution. For example, a significant result ( $P < \alpha$  level) rejects the hypothesis that the two matrices are from the same distribution.

**References**

Perkins, W., Tygert, M., & Ward, R. (2011). Computing the confidence levels for a root-mean-square test of goodness-of-fit. *Applied Mathematics and Computations*, 217(22), 9072-9084.

**Examples**

```
draws <- 17
observed <- matrix(c(.294, .176, .118, .411), nrow=2) * draws
expected <- matrix(c(.235, .235, .176, .353), nrow=2) * draws
ptw2011.gof.test(observed, expected) # not significant
```

---

<code>read.flexmirt</code>	<i>Read a flexMIRT PRM file</i>
----------------------------	---------------------------------

---

**Description**

Load the item parameters from a flexMIRT PRM file.

**Usage**

```
read.flexmirt(fname)
```

**Arguments**

<code>fname</code>	file name
--------------------	-----------

**Value**

a list of groups each with item parameters and the latent distribution

---

<code>rpf.1dim.fit</code>	<i>Calculate item and person Rasch fit statistics</i>
---------------------------	---

---

**Description**

Note: These statistics are only appropriate if all discrimination parameters are fixed equal and items are conditionally independent (see [ChenThissen1997](#)). A best effort is made to cope with missing data.

**Usage**

```
rpf.1dim.fit(spec, params, responses, scores, margin, group = NULL,
wh.exact = TRUE)
```

**Arguments**

<code>spec</code>	list of item models
<code>params</code>	matrix of item parameters, 1 per column
<code>responses</code>	persons in rows and items in columns
<code>scores</code>	model derived person scores
<code>margin</code>	for people 1, for items 2
<code>group</code>	spec, params, data, and scores can be provided in a list instead of as arguments
<code>wh.exact</code>	whether to use the exact Wilson-Hilferty transformation

## Details

Exact distributional properties of these statistics are unknown (Masters & Wright, 1997, p. 112). For details on the calculation, refer to Wright & Masters (1982, p. 100).

The Wilson-Hilferty transformation is biased for less than 25 items. Consider wh.exact=FALSE for less than 25 items.

## References

Masters, G. N. & Wright, B. D. (1997). The Partial Credit Model. In W. van der Linden & R. K. Kambleton (Eds.), *Handbook of modern item response theory* (pp. 101-121). Springer.

Wilson, E. B., & Hilferty, M. M. (1931). The distribution of chi-square. *Proceedings of the National Academy of Sciences of the United States of America*, 17, 684-688.

Wright, B. D. & Masters, G. N. (1982). *Rating Scale Analysis*. Chicago: Mesa Press.

## Examples

```
data(kct)
responses <- kct.people[,paste("V",2:19, sep="")]
rownames(responses) <- kct.people$NAME
colnames(responses) <- kct.items$NAME
scores <- kct.people$MEASURE
params <- cbind(1, kct.items$MEASURE, logit(0), logit(1))
rownames(params) <- kct.items$NAME
items<-list()
items[1:18] <- rpf.drm()
params[,2] <- -params[,2]
rpf.1dim.fit(items, t(params), responses, scores, 2, wh.exact=TRUE)
```

---

rpf.1dim.moment	<i>Calculate cell central moments</i>
-----------------	---------------------------------------

---

## Description

Popular central moments include 2 (variance) and 4 (kurtosis).

## Usage

```
rpf.1dim.moment(spec, params, scores, m)
```

## Arguments

spec	list of item models
params	data frame of item parameters, 1 per row
scores	model derived person scores
m	which moment

**Value**

moment matrix

---

rpf.1dim.residual      *Calculate residuals*

---

**Description**

Calculate residuals

**Usage**

rpf.1dim.residual(spec, params, responses, scores)

**Arguments**

spec	list of item models
params	data frame of item parameters, 1 per row
responses	persons in rows and items in columns
scores	model derived person scores

**Value**

residuals

---

rpf.1dim.stdresidual      *Calculate standardized residuals*

---

**Description**

Calculate standardized residuals

**Usage**

rpf.1dim.stdresidual(spec, params, responses, scores)

**Arguments**

spec	list of item models
params	data frame of item parameters, 1 per row
responses	persons in rows and items in columns
scores	model derived person scores

**Value**

standardized residuals

---

rpf.dLL	<i>Item parameter derivatives</i>
---------	-----------------------------------

---

### Description

Evaluate the partial derivatives of the log likelihood with respect to each parameter at where with weight.

### Usage

```
rpf.dLL(m, param, where, weight)
```

### Arguments

m	item model
param	item parameters
where	location in the latent space
weight	per outcome weights (typically derived by observation)

### Details

It is not easy to write an example for this function. To evaluate the derivative, you need to sum the derivatives across a quadrature. You also need response outcome weights at each quadrature point. It is not anticipated that this function will be often used in R code. It's mainly to expose a C-level function for occasional debugging.

### Value

first and second order partial derivatives of the log likelihood evaluated at where. For p parameters, the first p values are the first derivative and the next  $p(p+1)/2$  columns are the lower triangle of the second derivative.

### See Also

The numDeriv package.

rpf.drm

*Create a dichotomous response model***Description**

For slope vector  $a$ , intercept  $c$ , pseudo-guessing parameter  $g$ , upper bound  $u$ , and latent ability vector  $\theta$ , the response probability function is

$$P(\text{pick} = 0|a, c, g, u, \theta) = 1 - P(\text{pick} = 1|a, c, g, u, \theta)$$

$$P(\text{pick} = 1|a, c, g, u, \theta) = g + (u - g) \frac{1}{1 + \exp(-(a\theta + c))}$$

**Usage**

```
rpf.drm(factors = 1, multidimensional = TRUE, poor = FALSE)
```

**Arguments**

factors	the number of factors
multidimensional	whether to use a multidimensional model. Defaults to TRUE.
poor	if TRUE, use the traditional parameterization of the 1d model instead of the slope-intercept parameterization

**Details**

The pseudo-guessing and upper bound parameter are specified in logit units (see [logit](#)).

For discussion on the choice of priors see Cai, Yang, and Hansen (2011, p. 246).

**Value**

an item model

**References**

Cai, L., Yang, J. S., & Hansen, M. (2011). Generalized Full-Information Item Bifactor Analysis. *Psychological Methods*, 16(3), 221-248.

**Examples**

```
spec <- rpf.drm()
rpf.prob(spec, rpf.rparam(spec), 0)
```



---

rpf.dTheta	<i>Item derivatives with respect to the location in the latent space</i>
------------	--

---

**Description**

Evaluate the partial derivatives of the response probability with respect to ability. See [rpf.info](http://rpf.info) for an application.

**Usage**

```
rpf.dTheta(m, param, where, dir)
```

**Arguments**

m	item model
param	item parameters
where	location in the latent distribution
dir	if more than 1 factor, a basis vector]

---

rpf.grm	<i>Create a graded response model</i>
---------	---------------------------------------

---

**Description**

For outcomes  $k$  in 0 to  $K$ , slope vector  $a$ , intercept vector  $c$ , and latent ability vector  $\theta$ , the response probability function is

$$P(\text{pick} = 0 | a, c, \theta) = 1 - P(\text{pick} = 1 | a, c_1, \theta)$$

$$P(\text{pick} = k | a, c, \theta) = \frac{1}{1 + \exp(-(a\theta + c_k))} - \frac{1}{1 + \exp(-(a\theta + c_{k+1}))}$$

$$P(\text{pick} = K | a, c, \theta) = \frac{1}{1 + \exp(-(a\theta + c_K))}$$

**Usage**

```
rpf.grm(outcomes = 2, factors = 1, multidimensional = TRUE)
```

**Arguments**

outcomes	The number of choices available
factors	the number of factors
multidimensional	whether to use a multidimensional model. Defaults to TRUE.

**Details**

The graded response model was designed for a item with a series of dependent parts where a higher score implies that easier parts of the item were surmounted. If there is any chance your polytomous item has independent parts then consider [rpf.nrm](#). If your categories cannot cross then the graded response model provides a little more information than the nominal model. Stronger a priori assumptions offer provide more power at the cost of flexibility.

**Value**

an item model

**Examples**

```
spec <- rpf.grm()
rpf.prob(spec, rpf.rparam(spec), 0)
```

---

rpf.id\_of

*Convert an rpf item model name to an ID*

---

**Description**

This is an internal function and should not be used.

**Usage**

```
rpf.id_of(name)
```

**Arguments**

name            name of the item model (string)

**Value**

the integer ID assigned to the given model

---

rpf.info	<i>Map an item model, item parameters, and person trait score into a information vector</i>
----------	---

---

**Description**

Map an item model, item parameters, and person trait score into a information vector

**Usage**

```
rpf.info(ii, ii.p, where, basis = 1)
```

**Arguments**

ii	an item model
ii.p	item parameters
where	the location in the latent distribution
basis	if more than 1 factor, a positive basis vector

**Value**

Fisher information

**References**

Dodd, B. G., De Ayala, R. J. & Koch, W. R. (1995). Computerized adaptive testing with polytomous items. *Applied psychological measurement* 19(1), 5-22.

**Examples**

```
i1 <- rpf.drm()
i1.p <- c(.6,1,.1,.95)
theta <- seq(0,3,.05)
plot(theta, rpf.info(i1, i1.p, t(theta)), type="l")
```

---

rpf.lmp	<i>Create logistic function of a monotonic polynomial (LMP) model</i>
---------	---

---

**Description**

This model is a dichotomous response model originally proposed by Liang (2007) and is implemented using the parameterization by Falk & Cai (in press).

**Usage**

```
rpf.lmp(k = 0, multidimensional = FALSE)
```

### Arguments

- k** a non-negative integer that controls the order of the polynomial ( $2k+1$ ) with a default of  $k=0$  (1st order polynomial = 2PL).
- multidimensional** whether to use a multidimensional model. Defaults to FALSE. The multidimensional version is not yet available.

### Details

The LMP model replaces the linear predictor part of the two-parameter logistic function with a monotonic polynomial,  $m(\theta, \omega, \xi, \alpha, \tau)$ ,

$$P(\text{pick} = 1 | \omega, \xi, \alpha, \tau, \theta) = \frac{1}{1 + \exp(-(\xi + m(\theta, \omega, \xi, \alpha, \tau)))}$$

where  $\alpha$  and  $\tau$  are vectors of length  $k$ .

The order of the polynomial is always odd and is controlled by the user specified non-negative integer,  $k$ . The model contains  $2+2*k$  parameters and are used as input to the `rpf.prob` function in the following order:  $\omega$  - the natural log of the slope of the item model when  $k=0$ ,  $\xi$  - the intercept,  $\alpha$  and  $\tau$  - two parameters that control bends in the polynomial. These latter parameters are repeated in the same order for models with  $k>1$ . For example, a  $k=2$  polynomial will have an item parameter vector of:  $\omega, \xi, \alpha_1, \tau_1, \alpha_2, \tau_2$ .

See Falk & Cai (in press) for more details as to how the polynomial is constructed. In general, the polynomial looks like the following, but coefficients,  $b$ , are not directly estimated, but are a function of the item parameters.

$$m(\theta) = \xi + b_1\theta + b_2\theta^2 + \dots + b_{2k+1}\theta^{2k+1}$$

At the lowest order polynomial ( $k=0$ ) the model reduces to the two-parameter logistic (2PL) model. However, parameterization of the slope parameter,  $\omega$ , is currently different than the 2PL (i.e., slope =  $\exp(\omega)$ ). This parameterization ensures that the response function is always monotonically increasing without requiring constrained optimization.

Please note that the functions implementing this item model may eventually be replaced or subsumed by an alternative item model. That is, backwards compatibility will not necessarily be guaranteed and this item model should be considered experimental until further notice.

For example, Falk & Cai present a polytomous item model derived from the generalized partial credit model that also uses a monotonic polynomial as the linear predictor, referred to as a GPC-MP item model. Since the GPC-MP reduces to the LMP when the number of categories is 2, this is a potential candidate for replacing the LMP item model. An alternative may include the retention of a dichotomous response model, but with a lower (and upper) asymptote that further reduces to the three-parameter logistic (or four-parameter logistic) item model when  $k=0$ . Finally, future versions may reparameterize  $\omega$ , or allow the option to release constraints on monotonicity. For instance, releasing constraints on  $\omega$  may be desirable in cases where the user wishes to have the option of a monotonically decreasing response function. Further releasing constraints on  $\tau$  would allow nonmonotonicity and would be equivalent to replacing the linear predictor with a polynomial.

**Value**

an item model

**References**

Falk, C. F., & Cai, L. (in press). Maximum marginal likelihood estimation of a monotonic polynomial generalized partial credit model with applications to multiple group analysis. *Psychometrika*. <http://dx.doi.org/10.1007/s11336-014-9428-7>

Liang (2007). *A semi-parametric approach to estimating item response functions*. Unpublished doctoral dissertation, Department of Psychology, The Ohio State University.

**Examples**

```
spec <- rpf.lmp(1) # 3rd order polynomial
theta<-seq(-3,3,.1)
p<-rpf.prob(spec, c(-.11,.37,.24,-.21),theta)

spec <- rpf.lmp(2) # 5th order polynomial
p<-rpf.prob(spec, c(.69,.71,-.5,-8.48,.52,-3.32),theta)
```

---

rpf.logprob	<i>Map an item model, item parameters, and person trait score into a probability vector</i>
-------------	---

---

**Description**

Note that in general,  $\exp(\text{rpf.logprob}(\cdot)) \neq \text{rpf.prob}(\cdot)$  because the range of logits is much wider than the range of probabilities due to limitations of floating point numerical precision.

**Usage**

```
rpf.logprob(m, param, theta)
```

**Arguments**

m	an item model
param	item parameters
theta	the trait score(s)

**Value**

a vector of probabilities. For dichotomous items, probabilities are returned in the order incorrect, correct. Although redundant, both incorrect and correct probabilities are returned in the dichotomous case for API consistency with polytomous item models.

**Examples**

```
i1 <- rpf.drm()
i1.p <- rpf.rparam(i1)
rpf.logprob(i1, c(i1.p), -1) # low trait score
rpf.logprob(i1, c(i1.p), c(0,1)) # average and high trait score
```

---

`rpf.mcm`*Create a multiple-choice response model*

---

**Description**

WARNING: This model is mostly not implemented.

**Usage**

```
rpf.mcm(outcomes = 2, numChoices = 5, factors = 1)
```

**Arguments**

outcomes	the number of possible outcomes
numChoices	the number of choices available
factors	the number of factors

**Details**

This function instantiates a multiple-choice response model.

**Value**

an item model

**Author(s)**

Jonathan Weeks <weeksjp@gmail.com>

---

rpf.mean.info	<i>Find the point where an item provides mean maximum information</i>
---------------	---

---

**Description**

This is a point estimate of the mean difficulty of items that do not offer easily interpretable parameters such as the Generalized PCM. Since the information curve may not be unimodal, this function integrates across the latent space.

**Usage**

```
rpf.mean.info(spec, param, grain = 0.1)
```

**Arguments**

spec	list of item specs
param	list or matrix of item parameters
grain	the step size for numerical integration (optional)

**Details**

WARNING: This function is experimental and may disappear.

---

rpf.mean.info1	<i>Find the point where an item provides mean maximum information</i>
----------------	---

---

**Description**

WARNING: This function is experimental and may disappear.

**Usage**

```
rpf.mean.info1(spec, iparam, grain = 0.1)
```

**Arguments**

spec	an item spec
iparam	an item parameter vector
grain	the step size for numerical integration (optional)

---

`rpf.modify`*Create a similar item specification with the given number of factors*

---

**Description**

Create a similar item specification with the given number of factors

**Usage**

```
rpf.modify(m, factors)
```

**Arguments**

<code>m</code>	item model
<code>factors</code>	the number of factors/dimensions

**Examples**

```
s1 <- rpf.grm(factors=3)
rpf.rparam(s1)
s2 <- rpf.modify(s1, 1)
rpf.rparam(s2)
```

---

`rpf.nrm`*Create a nominal response model*

---

**Description**

This function instantiates a nominal response model.

**Usage**

```
rpf.nrm(outcomes = 3, factors = 1, T.a = "trend", T.c = "trend")
```

**Arguments**

<code>outcomes</code>	The number of choices available
<code>factors</code>	the number of factors
<code>T.a</code>	the T matrix for slope parameters
<code>T.c</code>	the T matrix for intercept parameters



**Details**

The transformation matrices  $T_a$  and  $T_c$  are chosen by the analyst and not estimated. The  $T$  matrices must be invertible square matrices of size `outcomes-1`. As a shortcut, either  $T$  matrix can be specified as "trend" for a Fourier basis or as "id" for an identity basis. The response probability function is

$$a = T_a \alpha$$

$$c = T_c \gamma$$

$$P(\text{pick} = k | s, a_k, c_k, \theta) = C \frac{1}{1 + \exp(- (s\theta a_k + c_k))}$$

where  $a_k$  and  $c_k$  are the result of multiplying two vectors of free parameters  $\alpha$  and  $\gamma$  by fixed matrices  $T_a$  and  $T_c$ , respectively;  $a_0$  and  $c_0$  are fixed to 0 for identification; and  $C$  is a normalizing factor to ensure that  $\sum_k P(\text{pick} = k) = 1$ .

**Value**

an item model

**References**

Thissen, D., Cai, L., & Bock, R. D. (2010). The Nominal Categories Item Response Model. In M. L. Nering & R. Ostini (Eds.), *Handbook of Polytomous Item Response Theory Models* (pp. 43–75). Routledge.

**Examples**

```
spec <- rpf.nrm()
rpf.prob(spec, rpf.rparam(spec), 0)
# typical parameterization for the Generalized Partial Credit Model
gpcm <- function(outcomes) rpf.nrm(outcomes, T.c=lower.tri(diag(outcomes-1),TRUE) * -1)
spec <- gpcm(4)
rpf.prob(spec, rpf.rparam(spec), 0)
```

---

`rpf.numParam`

*Length of the item parameter vector*

---

**Description**

Length of the item parameter vector

**Usage**

```
rpf.numParam(m)
```

**Arguments**

`m` item model

**Examples**

```
rpf.numParam(rpf.grm(outcomes=3))
rpf.numParam(rpf.nrm(outcomes=3))
```

---

rpf.numSpec	<i>Length of the item model vector</i>
-------------	--

---

**Description**

Length of the item model vector

**Usage**

```
rpf.numSpec(m)
```

**Arguments**

m	item model
---	------------

**Examples**

```
rpf.numSpec(rpf.grm(outcomes=3))
rpf.numSpec(rpf.nrm(outcomes=3))
```

---

rpf.ogive	<i>The ogive constant</i>
-----------	---------------------------

---

**Description**

The ogive constant can be multiplied by the discrimination parameter to obtain a response curve very similar to the Normal cumulative distribution function (Haley, 1952; Molenaar, 1974). Recently, Savalei (2006) proposed a new constant of 1.749 based on Kullback-Leibler information.

**Usage**

```
rpf.ogive
```

**Format**

An object of class `numeric` of length 1.

**Details**

In recent years, the logistic has grown in favor, and therefore, this package does not offer any special support for this transformation (Baker & Kim, 2004, pp. 14-18).

## References

- Camilli, G. (1994). Teacher's corner: Origin of the scaling constant  $d=1.7$  in Item Response Theory. *Journal of Educational and Behavioral Statistics*, 19(3), 293-295.
- Baker & Kim (2004). *Item Response Theory: Parameter Estimation Techniques*. Marcel Dekker, Inc.
- Haley, D. C. (1952). *Estimation of the dosage mortality relationship when the dose is subject to error* (Technical Report No. 15). Stanford University Applied Mathematics and Statistics Laboratory, Stanford, CA.
- Molenaar, W. (1974). De logistische en de normale kromme [The logistic and the normal curve]. *Nederlands Tijdschrift voor de Psychologie* 29, 415-420.
- Savalei, V. (2006). Logistic approximation to the normal: The KL rationale. *Psychometrika*, 71(4), 763-767.

---

rpf.paramInfo

*Retrieve a description of the given parameter*

---

## Description

Retrieve a description of the given parameter

## Usage

```
rpf.paramInfo(m, num = NULL)
```

## Arguments

m	item model
num	vector of parameters (defaults to all)

## Value

a list containing the type, upper bound, and lower bound

## Examples

```
rpf.paramInfo(rpf.drm())
```

---

rpf.prob	<i>Map an item model, item parameters, and person trait score into a probability vector</i>
----------	---

---

### Description

Map an item model, item parameters, and person trait score into a probability vector

### Usage

```
rpf.prob(m, param, theta)
```

### Arguments

m	an item model
param	item parameters
theta	the trait score(s)

### Value

a vector of probabilities. For dichotomous items, probabilities are returned in the order incorrect, correct. Although redundant, both incorrect and correct probabilities are returned in the dichotomous case for API consistency with polytomous item models.

### Examples

```
i1 <- rpf.drm()
i1.p <- rpf.rparam(i1)
rpf.prob(i1, c(i1.p), -1) # low trait score
rpf.prob(i1, c(i1.p), c(0,1)) # average and high trait score
```

---

rpf.rescale	<i>Rescale item parameters</i>
-------------	--------------------------------

---

### Description

Adjust item parameters for changes in mean and covariance of the latent distribution.

### Usage

```
rpf.rescale(m, param, mean, cov)
```

**Arguments**

m	item model
param	item parameters
mean	vector of means
cov	covariance matrix

**Examples**

```
spec <- rpf.grm()
p1 <- rpf.rparam(spec)
testPoint <- rnorm(1)
move <- rnorm(1)
cov <- as.matrix(rlnorm(1))
Icov <- solve(cov)
padj <- rpf.rescale(spec, p1, move, cov)
pr1 <- rpf.prob(spec, padj, (testPoint-move) %*% Icov)
pr2 <- rpf.prob(spec, p1, testPoint)
abs(pr1 - pr2) < 1e9
```

---

rpf.rparam	<i>Generates item parameters</i>
------------	----------------------------------

---

**Description**

This function generates random item parameters. The version argument is available if you are writing a test that depends on reproducible random parameters (using `set.seed`).

**Usage**

```
rpf.rparam(m, version = 2L)
```

**Arguments**

m	an item model
version	the version of random parameters

**Value**

item parameters

**Examples**

```
i1 <- rpf.drm()
rpf.rparam(i1)
```

rpf.sample

*Randomly sample response patterns given a list of items***Description**

Returns a random sample of response patterns given a list of item models and parameters. If `grp` is given then `theta`, `items`, `params`, `mean`, and `cov` can be omitted.

**Usage**

```
rpf.sample(theta, items, params, ..., prefix = "i", mean = NULL,
           cov = NULL, mcar = 0, grp = NULL)
```

**Arguments**

<code>theta</code>	either a vector (for 1 dimension) or a matrix (for >1 dimension) of person abilities or the number of response patterns to generate randomly
<code>items</code>	a list of item models
<code>params</code>	a list or matrix of item parameters. If omitted, random item parameters are generated for each item model.
<code>...</code>	Not used. Forces remaining arguments to be specified by name.
<code>prefix</code>	Column names are taken from <code>param</code> or <code>items</code> . If no column names are available, some will be generated using the given prefix.
<code>mean</code>	mean vector of latent distribution (optional)
<code>cov</code>	covariance matrix of latent distribution (optional)
<code>mcar</code>	proportion of generated data to set to NA (missing completely at random)
<code>grp</code>	a list with <code>spec</code> , <code>param</code> , <code>mean</code> , and <code>cov</code>

**Value**

Returns a data frame of response patterns

**See Also**

[sample](#)

**Examples**

```
# 1 dimensional items
i1 <- rpf.drm()
i1.p <- rpf.rparam(i1)
i2 <- rpf.nrm(outcomes=3)
i2.p <- rpf.rparam(i2)
rpf.sample(5, list(i1,i2), list(i1.p, i2.p))
```

---

science	<i>Liking for Science dataset</i>
---------	-----------------------------------

---

**Description**

These data are from Wright & Masters (1982, p. 18).

**Details**

All items were fit to a 3 category Partial Credit Model (PCM) using Ministep 3.75.0.

**References**

Wright, B. D. & Masters, G. N. (1982). *Rating Scale Analysis*. Chicago: Mesa Press.

**Examples**

```
data(science)
```

---

SitemFit	<i>Compute the S fit statistic for a set of items</i>
----------	---

---

**Description**

Runs `SitemFit1` for every item and accumulates the results.

**Usage**

```
SitemFit(grp, ..., method = "pearson", log = TRUE, qwidth = 6,
         qpoints = 49L, alt = FALSE, omit = 0L, .twotier = TRUE,
         .parallel = TRUE)
```

**Arguments**

<code>grp</code>	a list with spec, param, mean, cov, data, and the free variable pattern
<code>...</code>	Not used. Forces remaining arguments to be specified by name.
<code>method</code>	whether to use a pearson or rms test
<code>log</code>	whether to return pvalues in log units
<code>qwidth</code>	the positive width of the quadrature in Z units
<code>qpoints</code>	the number of quadrature points
<code>alt</code>	whether to include the item of interest in the denominator
<code>omit</code>	number of items to omit (a single number) or a list of the length the number of items
<code>.twotier</code>	whether to enable the two-tier optimization
<code>.parallel</code>	whether to take advantage of multiple CPUs (default TRUE)

**Value**

a list of output from [SitemFit1](#)

**Examples**

```
grp <- list(spec=list())
grp$spec[1:20] <- rpf.grm()
grp$param <- sapply(grp$spec, rpf.rparam)
colnames(grp$param) <- paste("i", 1:20, sep="")
grp$mean <- 0
grp$cov <- diag(1)
grp$free <- grp$param != 0
grp$data <- rpf.sample(500, grp=grp)
SitemFit(grp)
```

---

SitemFit1

*Compute the S fit statistic for 1 item*

---

**Description**

Implements the Kang & Chen (2007) polytomous extension to S statistic of Orlando & Thissen (2000). Rows with missing data are ignored, but see the omit option.

**Usage**

```
SitemFit1(grp, item, free = 0, ..., method = "pearson", log = TRUE,
          qwidth = 6, qpoints = 49L, alt = FALSE, omit = 0L, .twotier = TRUE)
```

**Arguments**

grp	a list with spec, param, mean, cov, and data
item	the item of interest
free	the number of free parameters involved in estimating the item (to adjust the df)
...	Not used. Forces remaining arguments to be specified by name.
method	whether to use a pearson or rms test
log	whether to return pvalues in log units
qwidth	the positive width of the quadrature in Z units
qpoints	the number of quadrature points
alt	whether to include the item of interest in the denominator
omit	number of items to omit or a character vector with the names of the items to omit when calculating the observed and expected sum-score tables
.twotier	whether to enable the two-tier optimization



**Details**

This statistic is good at finding a small number of misfitting items among a large number of well fitting items. However, be aware that misfitting items can cause other items to misfit.

Observed tables cannot be computed when data is missing. Therefore, you can optionally omit items with the greatest number of responses missing relative to the item of interest.

Pearson is slightly more powerful than RMS in most cases I examined.

Setting `alt` to TRUE causes the tables to match published articles. However, the default setting of FALSE probably provides slightly more power when there are less than 10 items.

The name of the test, "S", probably stands for sum-score.

**References**

Kang, T. and Chen, T. T. (2007). An investigation of the performance of the generalized S-Chisq item-fit index for polytomous IRT models. ACT Research Report Series.

Orlando, M. and Thissen, D. (2000). Likelihood-Based Item-Fit Indices for Dichotomous Item Response Theory Models. *Applied Psychological Measurement*, 24(1), 50-64.

---

stripData	<i>Strip data and scores from an IFA group</i>
-----------	--

---

**Description**

In addition, the `weightColumn` is reset to NULL.

**Usage**

```
stripData(grp)
```

**Arguments**

`grp`                    an IFA group

---

sumScoreEAP	<i>Compute the sum-score EAP table</i>
-------------	--

---

**Description**

Observed tables cannot be computed when data is missing. Therefore, you can optionally omit items with the greatest number of responses missing when conducting the distribution test.

**Usage**

```
sumScoreEAP(grp, ..., qwidth = 6, qpoints = 49L, .twotier = TRUE)
```

**Arguments**

grp	a list with spec, param, mean, and cov
...	Not used. Forces remaining arguments to be specified by name.
qwidth	positive width of quadrature in Z units
qpnts	number of quadrature points
.twotier	whether to enable the two-tier optimization

**Details**

When two-tier covariance structure is detected, EAP scores are only reported for primary factors. It is possible to compute EAP scores for specific factors, but it is not clear why this would be useful because they are conditional on the specific factor sum scores. Moreover, the algorithm to compute them efficiently has not been published yet (as of Jun 2014).

**Examples**

```
# see Thissen, Pommerich, Billeaud, & Williams (1995, Table 2)
spec <- list()
spec[1:3] <- rpf.grm(outcomes=4)

param <- matrix(c(1.87, .65, 1.97, 3.14,
                 2.66, .12, 1.57, 2.69,
                 1.24, .08, 2.03, 4.3), nrow=4)
# fix parameterization
param <- apply(param, 2, function(p) c(p[1], p[2:4] * -p[1]))

grp <- list(spec=spec, mean=0, cov=matrix(1,1,1), param=param)
sumScoreEAP(grp)
```

---

sumScoreEAPTest

---

*Conduct the sum-score EAP distribution test*


---

**Description**

Conduct the sum-score EAP distribution test

**Usage**

```
sumScoreEAPTest(grp, ..., qwidth = 6, qpnts = 49L, .twotier = TRUE)
```

**Arguments**

grp	a list with spec, param, mean, and cov
...	Not used. Forces remaining arguments to be specified by name.
qwidth	positive width of quadrature in Z units
qpnts	number of quadrature points
.twotier	whether to enable the two-tier optimization

## References

Li, Z., & Cai, L. (2012, July). Summed score likelihood based indices for testing latent variable distribution fit in Item Response Theory. Paper presented at the annual International Meeting of the Psychometric Society, Lincoln, NE. Retrieved from <http://www.cse.ucla.edu/downloads/files/SD2-final-4.pdf>

---

tabulateRows	<i>Tabulate data.frame rows</i>
--------------	---------------------------------

---

## Description

Like `tabulate` but entire rows are the unit of tabulation. The `data.frame` is not sorted, but must be sorted already.

## Usage

```
tabulateRows(observed)
```

## Arguments

`observed`          a sorted `data.frame` holding ordered factors in every column

## See Also

[orderCompletely](#)

## Examples

```
df <- as.data.frame(matrix(c(sample.int(2, 30, replace=TRUE)), 10, 3))
df <- df[orderCompletely(df),]
tabulateRows(df)
```

---

toFactorLoading	<i>Convert response function slopes to factor loadings</i>
-----------------	--

---

## Description

All slopes are divided by the ogive constant. Then the following transformation is applied to the slope matrix,

## Usage

```
toFactorLoading(slope, ogive = rpf.ogive)
```

**Arguments**

slope            a matrix with items in the columns and slopes in the rows  
ogive            the ogive constant (default rpf.ogive)

**Details**

$$\frac{\text{slope}}{[1 + \text{rowSums}(\text{slope}^2)]^{\frac{1}{2}}}$$

**Value**

a factor loading matrix with items in the rows and factors in the columns

**See Also**

[rpf.ogive](#)

---

toFactorThreshold	<i>Convert response function intercepts to factor thresholds</i>
-------------------	--

---

**Description**

Convert response function intercepts to factor thresholds

**Usage**

```
toFactorThreshold(intercept, slope, ogive = rpf.ogive)
```

**Arguments**

intercept        a matrix with items in the columns and intercepts in the rows  
slope            a matrix with items in the columns and slopes in the rows  
ogive            the ogive constant (default rpf.ogive)

**Value**

a factor threshold matrix with items in the columns and factor thresholds in the rows

---

write.flexmirt	<i>Write a flexMIRT PRM file</i>
----------------	----------------------------------

---

**Description**

Formats item parameters in the way that flexMIRT expects to read them. Use [read.flexmirt](#) to see what shape the groups parameter of this function should take.

**Usage**

```
write.flexmirt(groups, file = NULL, fileEncoding = "")
```

**Arguments**

groups	a list of groups each with items and latent parameters
file	the destination file name
fileEncoding	how to encode the text file (optional)

**Details**

NOTE: Support for the graded response model may not be complete.

# Index

- \*Topic **datasets**
  - rpf.ogive, 34
- \*Topic **data**
  - kct, 13
  - LSAT6, 14
  - LSAT7, 15
  - science, 39
- \$.rpf.base-method (Class rpf.base), 8
- \$<-, rpf.base-method (Class rpf.base), 8
- An introduction, 3
- An introduction-package (An introduction), 3
- as.IFAGroup, 4
- bestToOmit, 5
- chen.thissen.1997 (ChenThissen1997), 6
- ChenThissen1997, 6, 20
- Class rpf.1dim, 7
- Class rpf.1dim.drm, 7
- Class rpf.1dim.graded, 7
- Class rpf.1dim.grm, 7
- Class rpf.1dim.lmp, 8
- Class rpf.base, 8
- Class rpf.mdim, 8
- Class rpf.mdim.drm, 8
- Class rpf.mdim.graded, 8
- Class rpf.mdim.grm, 9
- Class rpf.mdim.mcm, 9
- Class rpf.mdim.nrm, 9
- compressDataFrame, 9
- crosstabTest, 10
- EAPscores, 4, 10
- expandDataFrame, 11
- fromFactorLoading, 12
- fromFactorThreshold, 12
- itemOutcomeBySumScore, 13
- kct, 13
- logit, 14, 24
- LSAT6, 14
- LSAT7, 15
- multinomialFit, 15
- observedSumScore, 16
- omitItems, 17
- omitMostMissing, 17
- orderCompletely, 18, 43
- ordinal.gamma, 6, 18
- ptw2011.gof.test, 19
- read.flexmirt, 20, 45
- rpf.1dim-class (Class rpf.1dim), 7
- rpf.1dim.drm-class (Class rpf.1dim.drm), 7
- rpf.1dim.fit, 20
- rpf.1dim.graded-class (Class rpf.1dim.graded), 7
- rpf.1dim.grm-class (Class rpf.1dim.grm), 7
- rpf.1dim.lmp-class (Class rpf.1dim.lmp), 8
- rpf.1dim.moment, 21
- rpf.1dim.residual, 22
- rpf.1dim.stdresidual, 22
- rpf.base-class (Class rpf.base), 8
- rpf.dLL, 23
- rpf.dLL, rpf.base, numeric, NULL, numeric-method (rpf.dLL), 23
- rpf.dLL, rpf.base, numeric, numeric, numeric-method (rpf.dLL), 23
- rpf.drm, 4, 24
- rpf.dTheta, 25
- rpf.dTheta, rpf.base, numeric, matrix, numeric-method (rpf.dTheta), 25

- rpf.dTheta, rpf.base, numeric, numeric, numeric-method (rpf.paramInfo), 35  
(rpf.dTheta), 25
- rpf.grm, 4, 25
- rpf.id\_of, 26
- rpf.info, 25, 27
- rpf.lmp, 27
- rpf.logprob, 29
- rpf.logprob, rpf.1dim, numeric, matrix-method  
(rpf.logprob), 29
- rpf.logprob, rpf.1dim, numeric, numeric-method  
(rpf.logprob), 29
- rpf.logprob, rpf.mdim, numeric, matrix-method  
(rpf.logprob), 29
- rpf.logprob, rpf.mdim, numeric, NULL-method  
(rpf.logprob), 29
- rpf.logprob, rpf.mdim, numeric, numeric-method  
(rpf.logprob), 29
- rpf.mcm, 30
- rpf.mdim-class (Class rpf.mdim), 8
- rpf.mdim.drm-class (Class  
rpf.mdim.drm), 8
- rpf.mdim.graded-class (Class  
rpf.mdim.graded), 8
- rpf.mdim.grm-class (Class  
rpf.mdim.grm), 9
- rpf.mdim.mcm-class (Class  
rpf.mdim.mcm), 9
- rpf.mdim.nrm-class (Class  
rpf.mdim.nrm), 9
- rpf.mean.info, 31
- rpf.mean.info1, 31
- rpf.modify, 32
- rpf.modify, rpf.mdim.drm, numeric-method  
(rpf.modify), 32
- rpf.modify, rpf.mdim.graded, numeric-method  
(rpf.modify), 32
- rpf.modify, rpf.mdim.nrm, numeric-method  
(rpf.modify), 32
- rpf.nrm, 4, 26, 32
- rpf.numParam, 33
- rpf.numParam, rpf.base-method  
(rpf.numParam), 33
- rpf.numSpec, 34
- rpf.numSpec, rpf.base-method  
(rpf.numSpec), 34
- rpf.ogive, 4, 34, 44
- rpf.paramInfo, 35
- rpf.paramInfo, rpf.base-method  
(rpf.paramInfo), 35
- rpf.prob, 36
- rpf.prob, rpf.1dim, numeric, matrix-method  
(rpf.prob), 36
- rpf.prob, rpf.1dim, numeric, numeric-method  
(rpf.prob), 36
- rpf.prob, rpf.1dim.grm, numeric, numeric-method  
(rpf.prob), 36
- rpf.prob, rpf.base, data.frame, numeric-method  
(rpf.prob), 36
- rpf.prob, rpf.base, matrix, matrix-method  
(rpf.prob), 36
- rpf.prob, rpf.base, matrix, numeric-method  
(rpf.prob), 36
- rpf.prob, rpf.mdim, numeric, matrix-method  
(rpf.prob), 36
- rpf.prob, rpf.mdim, numeric, NULL-method  
(rpf.prob), 36
- rpf.prob, rpf.mdim, numeric, numeric-method  
(rpf.prob), 36
- rpf.prob, rpf.mdim.grm, numeric, matrix-method  
(rpf.prob), 36
- rpf.prob, rpf.mdim.grm, numeric, numeric-method  
(rpf.prob), 36
- rpf.prob, rpf.mdim.mcm, numeric, matrix-method  
(rpf.prob), 36
- rpf.prob, rpf.mdim.nrm, numeric, matrix-method  
(rpf.prob), 36
- rpf.rescale, 36
- rpf.rescale, rpf.base, numeric, numeric, matrix-method  
(rpf.rescale), 36
- rpf.rparam, 4, 37
- rpf.rparam, rpf.1dim.drm-method  
(rpf.rparam), 37
- rpf.rparam, rpf.1dim.graded-method  
(rpf.rparam), 37
- rpf.rparam, rpf.1dim.lmp-method  
(rpf.rparam), 37
- rpf.rparam, rpf.mdim.drm-method  
(rpf.rparam), 37
- rpf.rparam, rpf.mdim.graded-method  
(rpf.rparam), 37
- rpf.rparam, rpf.mdim.mcm-method  
(rpf.rparam), 37
- rpf.rparam, rpf.mdim.nrm-method  
(rpf.rparam), 37
- rpf.sample, 38
- rpf\_dLL\_wrapper (rpf.dLL), 23

rpf\_dTheta\_wrapper (rpf.dTheta), 25  
rpf\_logprob\_wrapper (rpf.logprob), 29  
rpf\_numParam\_wrapper (rpf.numParam), 33  
rpf\_numSpec\_wrapper (rpf.numSpec), 34  
rpf\_paramInfo\_wrapper (rpf.paramInfo),  
35  
rpf\_prob\_wrapper (rpf.prob), 36  
rpf\_rescale\_wrapper (rpf.rescale), 36

sample, 38  
science, 39  
sfif (science), 39  
sfpf (science), 39  
sfsf (science), 39  
sfixf (science), 39  
SitemFit, 39  
SitemFit1, 39, 40, 40  
stripData, 41  
sumScoreEAP, 41  
sumScoreEAPTest, 42

tabulateRows, 43  
toFactorLoading, 43  
toFactorThreshold, 44

write.flexmirt, 45