Package 'pcaPA'

July 2, 2014

Title Parallel Analysis for ordinal and numeric data using polychoric and Pearson correlations with S3 classes.

Version 1.2

Date 2013-23-20

Author Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>.

Maintainer Carlos A. Arias <carias@icfes.gov.co>

Depends R (>= 3.0.0), polycor, ltm, stats, ggplot2, mc2d

Description A set of functions to perform parallel analysis for principal components analysis intended mainly for large data sets. It performs a parallel analysis of continuous, ordered (including dichotomous/binary as a special case) or mixed type of data associated with a principal components analysis. Polychoric correlations among ordered variables, Pearson correlations among continuous variables and polyserial correlation between mixed type variables (one ordered and one continuous) are used. Whenever the use of polyserial or polychoric correlations yields a non positive definite correlation matrix, the resulting matrix is transformed into the nearest positive definite matrix.

License GPL (>= 2)

NeedsCompilation yes

Repository CRAN

Date/Publication 2013-12-21 00:15:37

R topics documented:

`CalculatePABinary` *Parallel Analysis for Dichotomous Data.*

Description

Obtains a parallel analysis for dichotomous data.

Usage

```
CalculatePABinary(dataMatrix, percentiles = 0.99,nReplicates = 200,
                  use = "complete.obs", algorithm = "polycor")
```

Arguments

`dataMatrix`	`matrix` or `data.frame` of binary or dichotomous variables.
`percentiles`	vector of percentiles to report.
`nReplicates`	number of simulations to produce for estimating the eigenvalues distribution under independence.
`use`	Missing value handling method: If `"complete.obs"`, remove observations with any missing data; if `"pairwise.complete.obs"`, compute each correlation using all observations with valid data for that pair of variables.
`algorithm`	string specifying the correlation estimation algorithm. Polychoric correlation estimation method: `"polycor"` for estimates using the polycor package, `"polychoric"` for estimates using the `C++` function `Cpolychoric`.

Value

Returns a `list` object with the following:

`observed`	`data.frame` containing the observed eigenvalues.
`percentiles`	`data.frame` containing the estimated percentiles of the eigenvalues distribution under independence.
`simulatedEigenValues`	`data.frame` containing the simulated eigenvalues under independence.

Note

This is an auxiliary function for the "PA" function.

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

See Also

CalculatePAOrdered, CalculatePAContinuous, CalculatePAMixed, PA, quantile.PA

Examples

```
# # NOT RUN
# # Run Parallel Analysis for binary data conforming to the Rasch model
# # using the polycor package
# data(simRaschData)
# binaryRaschPA <- PA(simRaschData, percentiles = c(0.95, 0.99),
#                     nReplicates = 200, type = "binary")
# print(binaryRaschPA)

# # Run Parallel Analysis for binary data conforming to the Rasch model
# # using the Cpolychoric C++ function
data(simRaschData)
binaryRaschPA <- PA(simRaschData, percentiles = c(0.95, 0.99), nReplicates = 200,
                    type = "binary", algorithm = "polychoric")
print(binaryRaschPA)

# # NOT RUN
# # Run Parallel Analysis for binary data conforming to the 2PL model
# # using the polycor package
# data(sim2plData)
# binary2plPA <- PA(sim2plData, percentiles = c(0.95, 0.99), nReplicates = 200,
#                   type = "binary")
# print(binary2plPA)

# # Run Parallel Analysis for binary data conforming to the 2PL model
# # using the polychoric C++ function
data(sim2plData)
binary2plPA <- PA(sim2plData, percentiles = c(0.95, 0.99), nReplicates = 200,
                  type = "binary", algorithm = "polychoric")
print(binary2plPA)
```

CalculatePAContinuous *Parallel Analysis for continuous data.*

Description

Obtains a parallel analysis for continuous data.

Usage

```
CalculatePAContinuous(dataMatrix, percentiles = 0.99, nReplicates = 200,
                      use = "complete.obs", algorithm = "pearson")
```

Arguments

dataMatrix	matrix or data.frame of binary or dichotomous variables.
percentiles	vector of percentiles to report.
nReplicates	number of simulations to produce for estimating the eigenvalues distribution under independence.
use	Missing value handling method: If "complete.obs", remove observations with any missing data; if "pairwise.complete.obs", compute each correlation using all observations with valid data for that pair of variables.
algorithm	string specifying the estimation algorithm. In the case of continuous variables, only the Pearson correlations are used. Ignored if different to "pearson".

Value

Returns a list object with the following:

observed	data.frame containing the observed eigenvalues.
percentiles	data.frame containing the estimated percentiles of the eigenvalues distribution under independence.
simulatedEigenValues	data.frame containing the simulated eigenvalues under independence.

Note

This is an auxiliary function for the "PA" function.

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

See Also

CalculatePABinary, CalculatePAOrdered, CalculatePAMixed, PA

Examples

```
# # Run Parallel analyis of numeric data (Iris)
data(iris)
continuousPA <- PA(iris[, -5], percentiles = c(0.90, 0.99), nReplicates = 200,
                   type = "continuous", algorithm = "pearson")
print(continuousPA)
```

CalculatePAMixed *Parallel Analysis for numeric and ordered mixed data.*

Description

Obtains a parallel analysis for numeric and ordered mixed data.

Usage

```
CalculatePAMixed(dataMatrix, percentiles = 0.99, nReplicates = 200,
                 use = "complete.obs", algorithm = "polycor")
```

Arguments

dataMatrix	matrix or data.frame of ordered variables.
percentiles	vector of percentiles to report.
nReplicates	number of simulations to produce for estimating the eigenvalues distribution under independence.
use	Missing values handling method: If "complete.obs", remove observations with any missing data; if "pairwise.complete.obs", compute each correlation using all observations with valid data for that pair of variables.
algorithm	string specifying the correlation estimation method. In the case of mixed variables, only the polycor package is currently used, so this value must always be "polycor".

Value

Returns a list object with the following:

observed	data.frame containing the observed eigenvalues.
percentiles	data.frame containing the estimated percentiles of the eigenvalues distribution under independence.
simulatedEigenValues	data.frame containing the simulated eigenvalues under independence.

Note

This is an auxiliary function for the "PA" function.

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

See Also

CalculatePABinary, CalculatePAContinuous, CalculatePAOrdered, PA

Examples

```
# # NOT RUN
# # Run Parallel analysis of mixed ordered and continuous data
# data(mixedScience)
# mixedPA <- PA(mixedScience, percentiles = c(0.90, 0.99),
#               nReplicates = 200, type = "mixed")
# print(mixedPA)
```

`CalculatePAOrdered` *Parallel Analysis for Ordered Data.*

Description

Obtains a parallel analysis for ordered data.

Usage

```
CalculatePAOrdered(dataMatrix, percentiles = 0.99, nReplicates = 200,
                   use = "complete.obs", algorithm = "polycor")
```

Arguments

`dataMatrix`	`matrix` or `data.frame` of ordered variables.
`percentiles`	vector of percentiles to report.
`nReplicates`	number of simulations to produce for estimating the eigenvalues distribution under independence.
`use`	Missing value handling method: If `"complete.obs"`, remove observations with any missing data; if `"pairwise.complete.obs"`, compute each correlation using all observations with valid data for that pair of variables.
`algorithm`	string specifying the correlation estimation algorithm. Polychoric correlation estimation method: `"polycor"` for estimates using the polycor package, `"polychoric"` for estimates using the `C++` function `Cpolychoric`.

Value

Returns a `list` object with the following:

`observed`	`data.frame` containing the observed eigenvalues.
`percentiles`	`data.frame` containing the estimated percentiles of the eigenvalues distribution under independence.
`simulatedEigenValues`	`data.frame` containing the simulated eigenvalues under independence.

Note

This is an auxiliary function for the `"PA"` function.

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

See Also

CalculatePABinary, CalculatePAContinuous, CalculatePAMixed, PA

Examples

```
# # NOT RUN
# # Run Parallel analysis for ordered polytomous data using the polycor package
# data(Science)
# Science[, ] <- lapply(Science, as.ordered)
# orderedPA <- PA(Science, percentiles = c(0.90, 0.99), nReplicates = 200,
#                 type = "ordered")
# print(orderedPA)

# # Run Parallel analysis for ordered polytomous data using
# # he polychoric C++ function
data(Science)
Science[, ] <- lapply(Science, as.ordered)
orderedPA   <- PA(Science, percentiles = c(0.90, 0.99), nReplicates = 200,
                  type = "ordered", algorithm = "polychoric")
print(orderedPA)
```

Check.PA *Verifies that an object belongs to the "PA" class.*

Description

Checks if an object is of class "PA" (Parallel analysis).

Usage

```
Check.PA(PA)
```

Arguments

PA	An object to be checked for class "PA".

Value

Returns TRUE if input object if of class "PA" and has all the necessary components. Returns FALSE otherwise.

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

See Also

PA, print.PA, coef.PA, CountEigen.PA, plot.PA, quantile.PA

Examples

```
# # Run Parallel Analysis for binary data conforming to the Rasch model
data(simRaschData)
binaryRaschPA <- PA(simRaschData, percentiles = c(0.95, 0.99), nReplicates = 200,
                    type = "binary", algorithm = "polychoric")
print(binaryRaschPA)

# # Check if binaryRaschPA is a PA object
Check.PA(binaryRaschPA)  # Should return TRUE
Check.PA(simRaschData)  # Should return FALSE

# # Run Parallel Analysis for binary data conforming to the 2PL model
data(sim2plData)
binary2plPA <- PA(sim2plData, percentiles = c(0.95, 0.99), nReplicates = 200,
                  type = "binary", algorithm = "polychoric")
print(binary2plPA)

# # Check if binary2plPA is a PA object
Check.PA(binary2plPA)  # Should return TRUE
Check.PA(simRaschData)  # Should return FALSE
```

---

coef.PA *Eigenvalue and percentile extraction of a "PA" object.*

Description

coef method for objects of class "PA", produced by PA.

Usage

```
## S3 method for class 'PA'
coef(object, ...)
```

Arguments

object	an object of class "PA"
...	not used

Value

An object of class "matrix" with the observed eigenvalues and the percentiles.

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

See Also

PA, print.PA, Check.PA, CountEigen.PA, plot.PA, quantile.PA

Examples

```
# # Run Parallel Analysis for binary data conforming to the Rasch model
data(simRaschData)
binaryRaschPA <- PA(simRaschData, percentiles = c(0.95, 0.99), nReplicates = 200,
                    type = "binary", algorithm = "polychoric")
print(binaryRaschPA)
binaryRaschPAEigenValues <- coef(binaryRaschPA)  # Save the matrix of observed
                                                 # eigenvalues and estimated
                                                 # eigenvalue percentiles
binaryRaschPAEigenValues
```

CountEigen.PA *Number of observed eigenvalues that exceed a given set of percentiles.*

Description

Counts the number of observed eigenvalues that exceed the given percentiles.

Usage

```
CountEigen.PA(PA, percentiles = NULL)
```

Arguments

PA	an object of class "PA".
percentiles	the percentiles that ought to be plotted, defaults to those in the object.

Value

A named numeric vector indicating the number of eigenvalues that are greater than the eigenvalues distribution percentiles under independence.

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

See Also

PA, print.PA, coef.PA, Check.PA, plot.PA, quantile.PA

Examples

```
# # Run Parallel Analysis for binary data conforming to the Rasch model
data(simRaschData)
binaryRaschPA <- PA(simRaschData, percentiles = c(0.95, 0.99), nReplicates = 200,
                    type = "binary", algorithm = "polychoric")
print(binaryRaschPA)

# # Number of retained factors
nComponents <- CountEigen.PA(binaryRaschPA, percentiles = .99)
nComponents["p99"]

# # Run Parallel Analysis for binary data conforming to the 2PL model
data(sim2plData)
binary2plPA <- PA(sim2plData, percentiles = c(0.95, 0.99), nReplicates = 200,
                  type = "binary", algorithm = "polychoric")
print(binary2plPA)

# # Number of retained factors
nComponents <- CountEigen.PA(binary2plPA, percentiles = .99)
nComponents["p99"]
```

---

mixedScience *Simulated data from a normal distribution added to the Science data set from package "ltm".*

Description

Mixed ordered (Science dataset from package "ltm" coming from the Consumer Protection and Perceptions of Science and Technology section of the 1992 Euro-Barometer Survey) and continuous (simulated from a normal distribution) data.

Usage

```
data(mixedScience)
```

Format

A data frame with 392 observations on the following 12 variables:

`Comfort` an ordered factor with levels `strongly disagree` < `disagree` < `agree` < `strongly agree`

`Environment` an ordered factor with levels `strongly disagree` < `disagree` < `agree` < `strongly agree`

`Work` an ordered factor with levels `strongly disagree` < `disagree` < `agree` < `strongly agree`

`Future` an ordered factor with levels `strongly disagree` < `disagree` < `agree` < `strongly agree`

`Technology` an ordered factor with levels `strongly disagree` < `disagree` < `agree` < `strongly agree`

`Industry` an ordered factor with levels `strongly disagree` < `disagree` < `agree` < `strongly agree`

`Benefit` an ordered factor with levels `strongly disagree` < `disagree` < `agree` < `strongly agree`

`X1` a numeric vector

`X2` a numeric vector

`X3` a numeric vector

`X4` a numeric vector

`X5` a numeric vector

References

Karlheinz, R. and Melich, A. (1992). Euro-Barometer 38.1: Consumer Protection and Perceptions of Science and Technology. INRA (Europe), Brussels. [computer file]

Rizopoulos, D. (2006). ltm: An R package for Latent Variable Modelling and Item Response Theory Analyses. Journal of Statistical Software, 17(5), 1–25.

Examples

```
# # NOT RUN
# data(mixedScience)
# mixedPA <- PA(mixedScience, percentiles = c(0.90, 0.99), nReplicates = 200,
#               type = "mixed")
# print(mixedPA)
```

`PA` *General function to perform parallel analysis of continuous, ordered or mixed type data.*

Description

Creates an object of class PA using one of the PA functions. It performs a parallel analysis of continuous, orderd (including dichotomous/binary as a special case) or mixed type of data associated with a principal components analysis. Polychoric correlations among ordered variables, Pearson correlations among continuous variables and polyserial correlation between mixed type variables (one ordered and one continuous) are used. Whenever the use of polyserial or polychoric correlations yields a non positive definite correlation matrix, the resulting matrix is transformed into the nearest positive definite matrix by `nearcor`.

Usage

```
PA(dataMatrix, percentiles = 0.99, nReplicates = 200, type = "continuous",
   use = "complete.obs", algorithm = "polycor")
```

Arguments

dataMatrix	matrix or data.frame of continuous numeric variables.
percentiles	vector of percentiles to report.
nReplicates	number of simulations to produce for estimating the eigenvalues distribution under independence.
type	Data type: "continuous" if the data is continuous. "binary" if the data is dichotomous. "ordered" if the data is ordinal. "mixed" if the data is mixed rodered and continuous.
use	Missing value handling method: If "complete.obs", remove observations with any missing data; if "pairwise.complete.obs", compute each correlation using all observations with valid data for that pair of variables.
algorithm	string specifying the correlation estimation algorithm. Polychoric correlation estimation method: "polycor" for estimates using the polycor package, "polychoric" for estimates using the C++ function polychoric. Pearson correlation estimation: "pearson", only used for continuous data.

Details

This function generates an object of (S3) class "PA" by using one of the four support functions (CalculatePAContinuous, CalculatePAOrdered, CalculatePABinary, CalculatePAMixed). As noted by Presaghi & Desimoni (2011), when using the parallel analysis approach to select the number of components to retain, observed and simulated correlation matrices ought to be of the same kind. The purporse with this package is to provide a more flexible framework to work with parallel anaylisis data. For the case where all variables are ordered polytomous the C++ function polychoric or the function "polychor" from package "polycor" may be used. Pearson correlation from "cor" function is used; finally, for sets that mix variables of both types, the "polycor" package is used. Furthermore, objects of class "PA" retain the eigenvalues for simulated correlation matrices allowing the user to avoid multiple simulations for changing certain parameters such as the specific qunatile used to decide on the retained number of components; also some missing observations may be handled by the "use" parameter.

Value

An object of class "PA" with the following:

observed	data.frame containing the observed eigenvalues.
percentiles	data.frame containing the estimated percentiles of the eigenvalues distribution under independence.
simulatedEigenValues	data.frame containing the simulated eigenvalues under independence.

Note

The algorithm "polychoric" is only implemented for binary/dichotomous and ordered polytomous data. When mixed type data are used only "polycor" is currently available.

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

References

Horn, J. L. (1965). A rationale and test for the number of factors in factor analysis. Psychometrika, 30, 179–185.

Glorfeld, L. W. (1995). An Improvement on Horn's Parallel Analysis Methodology for Selecting the Correct Number of Factors to Retain. Educational and Psychological Measurement, 55(3), 377–393.

See Also

CalculatePAContinuous, CalculatePAOrdered, CalculatePABinary, CalculatePAMixed, print.PA, plot.PA, coef.PA, quantile.PA, CountEigen.PA, Check.PA

Examples

```
# # NOT RUN
# # Run Parallel Analysis for binary data conforming to the Rasch model
# # using the polycor package
# data(simRaschData)
# binaryRaschPA <- PA(simRaschData, percentiles = c(0.95, 0.99), nReplicates = 200,
#                     type = "binary")
# print(binaryRaschPA)

# # Run Parallel Analysis for binary data conforming to the Rasch model
# # using the polychoric C++ function
data(simRaschData)
binaryRaschPA <- PA(simRaschData, percentiles = c(0.95, 0.99), nReplicates = 200,
                    type = "binary", algorithm = "polychoric")
print(binaryRaschPA)

# # NOT RUN
# # Run Parallel Analysis for binary data conforming to the 2PL model
# # using the polycor package
# data(sim2plData)
# binary2plPA <- PA(sim2plData, percentiles = c(0.95, 0.99), nReplicates = 200,
#                   type = "binary")
# print(binary2plPA)

# # Run Parallel Analysis for binary data conforming to the 2PL model
# # using the polychoric C++ function
data(sim2plData)
binary2plPA <- PA(sim2plData, percentiles = c(0.95, 0.99), nReplicates = 200,
                  type = "binary", algorithm = "polychoric")
print(binary2plPA)

# # NOT RUN
# # Run Parallel analysis for ordered polytomous data using the polycor package
# data(Science)
```

```
# Science[, ] <- lapply(Science, as.ordered)
# orderedPA <- PA(Science, percentiles = c(0.90, 0.99), nReplicates = 200,
#                 type = "ordered")
# print(orderedPA)

# # Run Parallel analysis for ordered polytomous data using the polychoric C++ function
data(Science)
Science[, ] <- lapply(Science, as.ordered)
orderedPA   <- PA(Science, percentiles = c(0.90, 0.99), nReplicates = 200,
                  type = "ordered", algorithm = "polychoric")
print(orderedPA)

# # NOT RUN
# # Run Parallel analysis of mixed ordered and continuous data
# data(mixedScience)
# mixedPA <- PA(mixedScience, percentiles = c(0.90, 0.99), nReplicates = 200,
#               type = "mixed")
# print(mixedPA)

# # Run Parallel analyis of numeric data (Iris)
data(iris)
continuousPA <- PA(iris[, -5], percentiles = c(0.90, 0.99), nReplicates = 200,
                   type = "continuous", algorithm = "pearson")
print(continuousPA)
```

`plot.PA` *Plot method for PA objects.*

Description

`plot` method for objects of class `PA`. Plots the scree plot for a `"PA"` object for the selected percentiles using `ggplot`.

Usage

```
## S3 method for class 'PA'
plot(x, percentiles = NULL, main = NULL, xlab = NULL, ylab = NULL,
      groupLabel = NULL, colour = TRUE, linetype = TRUE, observed = "Observed",
      percentile = "th percentile", position = "after", sep = "", ...)
```

Arguments

`x`	an object of class `"PA"`.
`percentiles`	The percentiles that ought to be plotted. Defaults to those in the PA object.
`main`	Graph title instead of default.
`xlab`	Label for `x` axis instead of default.
`ylab`	Label for `y` axis instead of default.
`groupLabel`	Legend box name instead of default.

colour	Logical indicating whether to identify the observed eigenvalues and percentiles by colour.
linetype	Logical indicating whether to identify the observed eigenvalues and percentiles by linetype.
observed	Label for the observed data, default is "observed"
percentile	Graph title instead of default.
position	Position for the percentile label. "after" will position the label after the percentile number. "before" will position the label before the percentile number
sep	Character string to separate the label from the percentiles number.
...	Not used.

Value

ggplot object for plotting the scree plot.

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

See Also

PA, print.PA, Check.PA, CountEigen.PA, coef.PA, quantile.PA

Examples

```
# # Run Parallel Analysis for binary data conforming to the Rasch model
# # using the polychoric C++ function
data(simRaschData)
binaryRaschPA <- PA(simRaschData, percentiles = c(0.95, 0.99), nReplicates = 200,
                    type = "binary", algorithm = "polychoric")
print(binaryRaschPA)
plot(binaryRaschPA, percentiles = 0.99, groupLabel = "") # Plots the scree-plot
                                                         # with the 99th percentile
```

---

print.PA *Print method for PA objects.*

Description

print method for objects of class "PA". Prints the observed eigenvalues and the percentiles as a matrix.

Usage

```
## S3 method for class 'PA'
print(x, digits = max(3, getOption("digits") - 3), observed = "Observed",
      percentile = "th percentile", position = "after", sep = "", ...)
```

Arguments

x	an object of class "PA".
digits	number of digits to print.
observed	Label for the observed data, default is "observed"
percentile	Graph title instead of default.
position	Position for the percentile label. "after" will position the label after the percentile number. "before" will position the label before the percentile number
sep	Character string to separate the label from the percentiles number.
...	Not used.

Value

NULL

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

See Also

PA, plot.PA, Check.PA, CountEigen.PA, coef.PA, quantile.PA

Examples

```
# # Run Parallel Analysis for binary data conforming to the Rasch model
# # using the polycor package
data(simRaschData)
binaryRaschPA <- PA(simRaschData, percentiles = c(0.95, 0.99), nReplicates = 200,
                    type = "binary", algorithm = "polychoric")
print(binaryRaschPA)
```

---

quantile.PA *Generate new quantiles based on given percentiles for a PA object.*

Description

Calculates arbitrary quantiles from the simulatedEigenValues in a "PA" object.

Usage

```
## S3 method for class 'PA'
quantile(x, percentiles = .99, ...)
```

Arguments

x	An object of class "PA".
percentiles	The new percentiles to replace the ones found in the "PA" object.
...	Not used.

Value

An object of class "PA" with the percentiles data.frame replaced by those from the percentiles argument.

Author(s)

Carlos A. Arias <carias@icfes.gov.co> and Victor H. Cervantes <vcervantes@icfes.gov.co>

See Also

PA, print.PA, Check.PA, CountEigen.PA, coef.PA, plot.PA

Examples

```
# # Using the polychoric C++ function
data(Science)
Science[, ] <- lapply(Science, as.ordered)
orderedPA   <- PA(Science, percentiles = c(0.90, 0.99), nReplicates = 200,
                  type = "ordered", algorithm = "polychoric")

print(orderedPA)  # Shows the eigenvalues as a matrix
head(orderedPA$simulatedEigenValues)  # Shows the first six iterations

# # Get different quantiles from a PA object
quantile(orderedPA, percentiles = c(0.1, 0.2, 0.5))
```

---

sim2plData *Simulated data conforming to the 2pl model.*

Description

Dichotomous data simulated from a 2pl model.

Usage

```
data(sim2plData)
```

Format

The format is: num [1:300, 1:15] 1 0 1 1 1 1 0 1 1 0 ...

Source

Data simulated using the eRm package.

Examples

```
data(sim2plData)
binary2plPA <- PA(sim2plData, percentiles = c(0.95, 0.99), nReplicates = 200,
                  type = "binary", algorithm = "polychoric")
print(binary2plPA)
```

---

simRaschData *Simulated data conforming to the Rasch Model.*

Description

Dichotomous data simulated from the Rasch model using package eRm.

Usage

```
data(simRaschData)
```

Format

Object whit class `data.frame` of 300 rows and 15 columns.

Index