

# Package ‘exametrika’

May 8, 2026

**Type** Package

**Title** Test Theory Analysis and Biclustering

**Version** 1.11.0

**Description** Implements comprehensive test data engineering methods as described in Shojima (2022, ISBN:978-9811699856). Provides statistical techniques for engineering and processing test data: Classical Test Theory (CTT) with reliability coefficients for continuous ability assessment; Item Response Theory (IRT) including Rasch, 2PL, and 3PL models with item/test information functions; Latent Class Analysis (LCA) for nominal clustering; Latent Rank Analysis (LRA) for ordinal clustering with automatic determination of cluster numbers; Biclustering methods including infinite relational models for simultaneous clustering of examinees and items without predefined cluster numbers; and Bayesian Network Models (BNM) for visualizing inter-item dependencies. Features local dependence analysis through LRA and biclustering, parameter estimation, dimensionality assessment, and network structure visualization for educational, psychological, and social science research.

**License** MIT + file LICENSE

**Language** en-US

**Encoding** UTF-8

**LazyData** true

**BuildVignettes** true

**URL** <https://kosugitti.github.io/exametrika/>,  
<https://github.com/kosugitti/exametrika>

**BugReports** <https://github.com/kosugitti/exametrika/issues>

**RoxygenNote** 7.3.3

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**Depends** R (>= 4.1.0)

**Imports** mvtnorm, igraph, Rcpp (>= 1.0.7)

**LinkingTo** Rcpp

**NeedsCompilation** yes

**Author** Koji Kosugi [aut, cre] (ORCID: <<https://orcid.org/0000-0001-5816-0099>>)

**Maintainer** Koji Kosugi <kosugitti@gmail.com>

**Repository** CRAN

**Date/Publication** 2026-04-15 05:40:02 UTC

## Contents

AlphaCoefficient . . . . .	4
AlphaIfDel . . . . .	5
asyprior . . . . .	5
Biclustering . . . . .	6
Biclustering_IRM . . . . .	10
BINET . . . . .	16
BiserialCorrelation . . . . .	19
BitRespPtn . . . . .	20
BNM . . . . .	20
BNM_GA . . . . .	22
BNM_PBIL . . . . .	24
calcFitIndices . . . . .	26
CCRR . . . . .	27
cr . . . . .	28
CSR . . . . .	29
CTT . . . . .	30
dataFormat . . . . .	31
Dimensionality . . . . .	32
DistractorAnalysis . . . . .	33
GridSearch . . . . .	34
GRM . . . . .	36
grm_iif . . . . .	37
grm_prob . . . . .	38
IIF2PLM . . . . .	39
IIF3PLM . . . . .	39
InterItemAnalysis . . . . .	40
IRM . . . . .	42
IRT . . . . .	43
ITBiserial . . . . .	44
ItemEntropy . . . . .	45
ItemFit . . . . .	47
ItemInformationFunc . . . . .	48
ItemLift . . . . .	48
ItemOdds . . . . .	49
ItemReport . . . . .	50
ItemStatistics . . . . .	52
ItemThreshold . . . . .	53

ItemTotalCorr . . . . .	54
J12S5000 . . . . .	56
J15S3810 . . . . .	56
J15S500 . . . . .	57
J20S400 . . . . .	57
J20S600 . . . . .	58
J21S300 . . . . .	58
J35S500 . . . . .	59
J35S5000 . . . . .	59
J35S515 . . . . .	60
J50S100 . . . . .	60
J5S10 . . . . .	61
J5S1000 . . . . .	61
JCRR . . . . .	62
JointSampleSize . . . . .	63
JSR . . . . .	64
LCA . . . . .	64
LDB . . . . .	67
LDLRA . . . . .	70
LDLRA_PBIL . . . . .	73
LD_param_est . . . . .	76
LogisticModel . . . . .	76
longdataFormat . . . . .	77
LRA . . . . .	78
maxParents_penalty . . . . .	83
MutualInformation . . . . .	84
nrs . . . . .	85
objective_function_IRT . . . . .	87
OmegaCoefficient . . . . .	87
passage . . . . .	88
percentile . . . . .	89
PhiCoefficient . . . . .	90
plot.exametrika . . . . .	91
polychoric . . . . .	95
PolychoricCorrelationMatrix . . . . .	96
polyserial . . . . .	97
print.exametrika . . . . .	98
PSD_item_params . . . . .	99
RaschModel . . . . .	100
ScoreReport . . . . .	100
slopeprior . . . . .	101
softmax . . . . .	102
sscore . . . . .	102
stanine . . . . .	103
StrLearningGA_BNM . . . . .	105
StrLearningPBIL_BNM . . . . .	105
StrLearningPBIL_LDLRA . . . . .	106
StudentAnalysis . . . . .	106

TestFit . . . . .	107
TestFitSaturated . . . . .	108
TestInformationFunc . . . . .	109
TestResponseFunc . . . . .	109
TestStatistics . . . . .	110
tetrachoric . . . . .	112
TetrachoricCorrelationMatrix . . . . .	113
ThreePLM . . . . .	114
TwoPLM . . . . .	114

<b>Index</b>	<b>116</b>
--------------	------------

---

AlphaCoefficient	<i>Alpha Coefficient</i>
------------------	--------------------------

---

### Description

This function computes Tau-Equivalent Measurement, also known as Cronbach's alpha coefficient, for a given data set.

### Usage

```
AlphaCoefficient(x, na = NULL, Z = NULL, w = NULL)
```

### Arguments

x	This should be a data matrix or a Covariance/Phi/Tetrachoric matrix.
na	This parameter identifies the numbers or characters that should be treated as missing values when 'x' is a data matrix.
Z	This parameter represents a missing indicator matrix. It is only needed if 'x' is a data matrix.
w	This parameter is an item weight vector. It is only required if 'x' is a data matrix.

### Value

For a correlation/covariance matrix input, returns a single numeric value representing the alpha coefficient. For a data matrix input, returns a list with three components:

**AlphaCov** Alpha coefficient calculated from covariance matrix

**AlphaPhi** Alpha coefficient calculated from phi coefficient matrix

**AlphaTetrachoric** Alpha coefficient calculated from tetrachoric correlation matrix

### References

Cronbach, L. J. (1951). Coefficient alpha and the internal structure of a test. *Psychometrika*, 16,297–334.

---

AlphaIfDel	<i>Alpha Coefficient if Item removed</i>
------------	--

---

**Description**

This function returns the alpha coefficient when the specified item is excluded.

**Usage**

```
AlphaIfDel(x, delItem = NULL, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

x	This should be a data matrix or a Covariance/Phi/Tetrachoric matrix.
delItem	Specify the item to be deleted. If NULL, calculations are performed for all cases.
na	This parameter identifies the numbers or characters that should be treated as missing values when 'x' is a data matrix.
Z	This parameter represents a missing indicator matrix. It is only needed if 'x' is a data matrix.
w	This parameter is an item weight vector. It is only required if 'x' is a data matrix.

---

asymprior	<i>Prior distribution function with guessing parameter</i>
-----------	--

---

**Description**

Prior distribution function with guessing parameter

**Usage**

```
asymprior(c, alp, bet)
```

**Arguments**

c	guessing parameter
alp	prior to be set
bet	prior to be set

**Description**

Performs biclustering, rankclustering, or their confirmatory variants on binary response data. These methods simultaneously cluster both examinees and items into homogeneous groups (or ordered ranks for rankclustering). The analysis reveals latent structures and patterns in the data by creating a matrix with rows and columns arranged to highlight block structures.

**Usage**

```
Biclustering(U, ...)  
  
## Default S3 method:  
Biclustering(U, na = NULL, Z = NULL, w = NULL, ...)  
  
## S3 method for class 'binary'  
Biclustering(  
  U,  
  ncls = 2,  
  nfls = 2,  
  method = "B",  
  conf = NULL,  
  mic = FALSE,  
  maxiter = 100,  
  verbose = TRUE,  
  beta1 = 1,  
  beta2 = 1,  
  ...  
)  
  
## S3 method for class 'nominal'  
Biclustering(  
  U,  
  ncls = 2,  
  nfls = 2,  
  conf = NULL,  
  mic = FALSE,  
  maxiter = 100,  
  verbose = TRUE,  
  alpha = 1,  
  ...  
)  
  
## S3 method for class 'ordinal'  
Biclustering(  
  U,  
  ncls = 2,  
  nfls = 2,  
  conf = NULL,  
  mic = FALSE,  
  maxiter = 100,  
  verbose = TRUE,  
  alpha = 1,  
  ...  
)
```

```

    U,
    ncls = 2,
    nflld = 2,
    method = "B",
    conf = NULL,
    mic = FALSE,
    maxiter = 100,
    verbose = TRUE,
    alpha = 1,
    ...
)

## S3 method for class 'rated'
Biclustering(
  U,
  ncls = 2,
  nflld = 2,
  method = "R",
  conf = NULL,
  maxiter = 100,
  verbose = TRUE,
  alpha = 1,
  ...
)

```

### Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
...	Additional arguments passed to specific methods.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.
ncls	Number of latent classes/ranks to identify (between 2 and 20).
nflld	Number of latent fields (item clusters) to identify.
method	Analysis method to use (character string): <ul style="list-style-type: none"> <li>• "B" or "Biclustering": Standard biclustering (default)</li> <li>• "R" or "Ranklustering": Ranklustering with ordered class structure</li> </ul>
conf	Confirmatory parameter for pre-specified field assignments. Can be either: <ul style="list-style-type: none"> <li>• A vector with items and corresponding fields in sequence</li> <li>• A field membership profile matrix (items x fields) with 0/1 values</li> <li>• NULL (default) for exploratory analysis where field memberships are estimated</li> </ul>

<code>mic</code>	Logical; if TRUE, forces Field Reference Profiles to be monotonically increasing. Default is FALSE.
<code>maxiter</code>	Maximum number of EM algorithm iterations. Default is 100.
<code>verbose</code>	Logical; if TRUE, displays progress during estimation. Default is TRUE.
<code>beta1</code>	Beta distribution parameter 1 for prior density of field reference matrix. Default is 1.
<code>beta2</code>	Beta distribution parameter 2 for prior density of field reference matrix. Default is 1.
<code>alpha</code>	Dirichlet distribution concentration parameter for prior density of field reference probabilities. Default is 1.

### Details

Biclustering simultaneously clusters both rows (examinees) and columns (items) of a data matrix. Unlike traditional clustering that groups either rows or columns, biclustering identifies submatrices with similar patterns. Rankclustering is a variant that imposes an ordinal structure on the classes, making it suitable for proficiency scaling.

The algorithm uses an Expectation-Maximization approach to iteratively estimate:

1. Field membership of items (which items belong to which fields)
2. Class/rank membership of examinees (which examinees belong to which classes)
3. Field Reference Profiles (probability patterns for each field-class combination)

The confirmatory option allows for pre-specified field assignments, which is useful when there is prior knowledge about item groupings or for testing hypothesized structures.

### Value

An object of class "exametrika" and "Biclustering" containing:

**model** Model type indicator (1 for biclustering, 2 for rankclustering)

**msg** A character string indicating the model type.

**mic** Logical value indicating whether monotonicity constraint was applied

**testlength** Number of items in the test

**nobs** Number of examinees in the dataset

**Nclass** Number of latent classes/ranks specified

**Nfield** Number of latent fields specified

**N\_Cycle** Number of EM iterations performed

**converge** Logical value indicating whether the algorithm converged within maxiter iterations

**LFD** Latent Field Distribution - counts of items assigned to each field

**LRD/LCD** Latent Rank/Class Distribution - counts of examinees assigned to each class/rank

**FRP** Field Reference Profile matrix - probability of correct response for each field-class combination

**FRPIndex** Field Reference Profile indices (Kumagai, 2007) — data.frame with 6 columns:

**Alpha** Maximum-slope location: class transition where the largest increase occurs

**A** Maximum slope: largest consecutive-class difference in the profile

**Beta** Location parameter: class whose FRP value is closest to 0.5

**B** FRP value at the Beta position

**Gamma** Non-monotonicity ratio: proportion of class transitions that decrease

**C** Monotonicity violation: sum of negative consecutive-class differences (C=0 means perfectly monotone)

For binary data, computed directly from the FRP matrix (correct response rates). For ordinal data, computed from normalized expected scores:  $(E[\text{score}]-1)/(\max Q-1)$ , which maps expected scores from  $[1, \max Q]$  to  $[0, 1]$  for comparability with the binary case. Not computed for nominal data (no natural category ordering).

**TRP** Test Reference Profile - expected score for examinees in each class/rank

**CMD/RMD** Class/Rank Membership Distribution - sum of membership probabilities across examinees

**FieldMembership** Matrix showing the probabilities of each item belonging to each field

**ClassMembership** Matrix showing the probabilities of each examinee belonging to each class/rank

**SmoothedMembership** Matrix of smoothed class membership probabilities after filtering

**FieldEstimated** Vector of the most likely field assignments for each item

**ClassEstimated** Vector of the most likely class/rank assignments for each examinee

**Students** Data frame containing membership probabilities and classification information for each examinee

**FieldAnalysis** Matrix showing field analysis results with item-level information

**TestFitIndices** Model fit indices for evaluating the quality of the clustering solution

**SOACflag** Logical flag indicating whether Strongly Ordinal Alignment Condition is satisfied

**WOACflag** Logical flag indicating whether Weakly Ordinal Alignment Condition is satisfied

## References

Shojima, K. (2012). Biclustering of binary data matrices using bilinear models. *Behaviormetrika*, 39(2), 161-178.

## Examples

```
# Perform Biclustering with Binary method (B)
# Analyze data with 5 fields and 6 classes
result.Bi <- Biclustering(J35S515, nfld = 5, ncls = 6, method = "B")

# Perform Biclustering with Rank method (R)
# Store results for further analysis and visualization
result.Rank <- Biclustering(J35S515, nfld = 5, ncls = 6, method = "R")

# Display the Bicluster Reference Matrix (BRM) as a heatmap
plot(result.Rank, type = "Array")

# Plot Field Reference Profiles (FRP) in a 2x3 grid
```

```

# Shows the probability patterns for each field
plot(result.Rank, type = "FRP", nc = 2, nr = 3)

# Plot Rank Membership Profiles (RMP) for students 1-9 in a 3x3 grid
# Shows posterior probability distribution of rank membership
plot(result.Rank, type = "RMP", students = 1:9, nc = 3, nr = 3)

# Example of confirmatory analysis with pre-specified fields
# Assign items 1-10 to field 1, 11-20 to field 2, etc.
field_assignments <- c(rep(1, 10), rep(2, 10), rep(3, 15))
result.Conf <- Biclustering(J35S515, nfld = 3, ncls = 5, conf = field_assignments)

# Perform Biclustering for nominal sample data()
# Analyze data with 5 fields and 6 classes
result.Bi <- Biclustering(J35S515, nfld = 5, ncls = 6, method = "B")

# Perform Biclustering for rated sample data
# Analyze data with 5 fields and 6 classes
result.Bi <- Biclustering(J35S5000, nfld = 5, ncls = 6, method = "R")

```

---

Biclustering\_IRM

*Biclustering with Infinite Relational Model*


---

## Description

This function performs Biclustering structure learning using the Infinite Relational Model (IRM) to automatically determine the optimal number of classes  $C$  and optimal number of fields  $F$ . It dispatches to the appropriate method based on the data type: binary, ordinal, or nominal. For binary data, see Section 7.8 in Shojima(2022). For nominal data, a Dirichlet-Multinomial collapsed Gibbs sampler is used.

## Usage

```

Biclustering_IRM(U, ...)

## Default S3 method:
Biclustering_IRM(U, na = NULL, Z = NULL, w = NULL, ...)

## S3 method for class 'binary'
Biclustering_IRM(
  U,
  Z = NULL,
  w = NULL,
  na = NULL,
  gamma_c = 1,
  gamma_f = 1,

```

```
    max_iter = 100,
    stable_limit = 5,
    minSize = 20,
    EM_limit = 20,
    seed = 123,
    verbose = TRUE,
    ...
)

## S3 method for class 'nominal'
Biclustering_IRM(
  U,
  gamma_c = 1,
  gamma_f = 1,
  alpha = 1,
  max_iter = 100,
  stable_limit = 5,
  minSize = 20,
  EM_limit = 20,
  seed = 123,
  verbose = TRUE,
  ...
)

## S3 method for class 'ordinal'
Biclustering_IRM(
  U,
  gamma_c = 1,
  gamma_f = 1,
  alpha = 1,
  mic = TRUE,
  max_iter = 100,
  stable_limit = 5,
  minSize = 20,
  EM_limit = 100,
  seed = 123,
  verbose = TRUE,
  ...
)

## S3 method for class 'rated'
Biclustering_IRM(
  U,
  gamma_c = 1,
  gamma_f = 1,
  alpha = 1,
  max_iter = 100,
  stable_limit = 5,
```

```

    minSize = 20,
    EM_limit = 20,
    seed = 123,
    verbose = TRUE,
    ...
)

```

## Arguments

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
...	Additional arguments passed to specific methods.
na	na argument specifies the numbers or characters to be treated as missing values.
Z	Z is a missing indicator matrix of the type matrix or data.frame
w	w is item weight vector
gamma_c	$\gamma_C$ is the hyperparameter of the CRP and represents the attractiveness of a new Class. As $\gamma_C$ increases, the student is more likely to be seated at a vacant class. The default is 1.
gamma_f	$\gamma_F$ is the hyperparameter of the CRP and represents the attractiveness of a new Field. The greater this value it more likely to be classified in the new field. The default is 1.
max_iter	A maximum iteration number of IRM process. The default is 100.
stable_limit	The IRM process exits the loop when the FRM stabilizes and no longer changes significantly. This option sets the maximum number of stable iterations, with a default of 5.
minSize	A value used for readjusting the number of classes.If the size of each class is less than <code>minSize</code> , the number of classes will be reduced. Note that this under limit of size is not used for either all correct or all incorrect class.
EM_limit	After IRM process, resizing the number of classes process will starts. This process using EM algorithm, <code>EM_limit</code> is the maximum number of iteration with default of 20.
seed	Random seed for reproducibility. When a numeric value is provided, <code>set.seed(seed)</code> is called before the Gibbs sampling begins, ensuring reproducible results. The default is 123, which guarantees deterministic output. Set to NULL to disable seed setting and let the results depend on the current state of the random number generator.
verbose	verbose output Flag. default is TRUE
alpha	Dirichlet distribution concentration parameter for the prior density of field reference probabilities (rated/nominal IRM). Must be positive. The default is 1.
mic	Logical; if TRUE, forces Field Reference Profiles to be monotonically increasing across classes (ordinal IRM only). Default is TRUE.

**Value**

An object of class "exametrika" containing the IRM results. See Biclustering\_IRM.binary or Biclustering\_IRM.nominal for details.

**nobs** Sample size. The number of rows in the dataset.

**msg** A character string indicating the model type.

**testlength** Length of the test. The number of items included in the test.

**n\_class** Optimal number of classes (new naming convention).

**n\_field** Optimal number of fields (new naming convention).

**em\_cycle** Number of EM algorithm iterations (new naming convention).

**Nclass** Optimal number of classes (deprecated, use n\_class).

**Nfield** Optimal number of fields (deprecated, use n\_field).

**EM\_Cycle** Number of EM algorithm iterations (deprecated, use em\_cycle).

**BRM** Bicluster Reference Matrix

**FRP** Field Reference Profile

**FRPIndex** Index of FRP includes the item location parameters B and Beta, the slope parameters A and Alpha, and the monotonicity indices C and Gamma.

**TRP** Test Reference Profile

**FMP** Field Membership Profile

**Students** Rank Membership Profile matrix. The s-th row vector of  $\hat{M}_R$ ,  $\hat{m}_R$ , is the rank membership profile of Student s, namely the posterior probability distribution representing the student's belonging to the respective latent classes. It also includes the rank with the maximum estimated membership probability, as well as the rank-up odds and rank-down odds.

**LRD** Latent Rank Distribution. see also [plot.exametrika](#)

**LFD** Latent Field Distribution. see also [plot.exametrika](#)

**RMD** Rank Membership Distribution.

**TestFitIndices** Overall fit index for the test. See also [TestFit](#)

For nominal data, the returned list includes:

**Q** Response matrix.

**Z** Missing indicator matrix.

**testlength** Number of items.

**nobs** Sample size.

**n\_class** Optimal number of classes.

**n\_field** Optimal number of fields.

**n\_cycle** Number of EM algorithm iterations.

**FRP** Field Reference Profile, a 3D array (nfld x ncls x maxQ).

**LFD** Latent Field Distribution.

**LCD** Latent Class Distribution.

**FieldMembership** Field membership probability matrix.  
**ClassMembership** Class membership probability matrix.  
**FieldEstimated** Estimated field assignment for each item.  
**ClassEstimated** Estimated class assignment for each student.  
**Students** Rank Membership Profile matrix with estimated class.  
**TestFitIndices** Overall fit index for the test.  
**log\_lik** Log-likelihood of the model.

For ordinal data, the returned list includes:

**Q** Response matrix.  
**Z** Missing indicator matrix.  
**testlength** Number of items.  
**nobs** Sample size.  
**n\_class** Optimal number of classes.  
**n\_field** Optimal number of fields.  
**n\_cycle** Number of EM algorithm iterations.  
**FRP** Field Reference Profile (BCRM), a 3D array (nfld x ncls x maxQ).  
**FRPIndex** Index of FRP includes the item location parameters B and Beta, the slope parameters A and Alpha, and the monotonicity indices C and Gamma.  
**TRP** Test Reference Profile.  
**BFRP** Bicluster Field Reference Profile (expected scores), a list with Weighted and Observed components.  
**LFD** Latent Field Distribution.  
**LCD** Latent Class Distribution.  
**FieldMembership** Field membership probability matrix.  
**ClassMembership** Class membership probability matrix.  
**FieldEstimated** Estimated field assignment for each item.  
**ClassEstimated** Estimated class assignment for each student.  
**Students** Rank Membership Profile matrix with estimated class.  
**TestFitIndices** Overall fit index for the test.  
**log\_lik** Log-likelihood of the model.  
**SOACflg** Logical; TRUE if Strongly Ordinal Alignment Condition is satisfied.  
**WOACflg** Logical; TRUE if Weakly Ordinal Alignment Condition is satisfied.

For rated data, the returned list includes:

**Q** Response matrix.  
**U** Binary correct/incorrect matrix.  
**Z** Missing indicator matrix.  
**testlength** Number of items.

**nobs** Sample size.

**n\_class** Optimal number of classes.

**n\_field** Optimal number of fields.

**n\_cycle** Number of EM algorithm iterations.

**FRP** Field Reference Profile (BCRM), a 3D array (nfld x ncls x maxQ).

**FieldFRP** Field Reference Profile based on binary correct rates (nfld x ncls).

**quasiFRP** Item-level correct response rate per class (nitems x ncls).

**FRPIndex** Index of FRP includes the item location parameters B and Beta, the slope parameters A and Alpha, and the monotonicity indices C and Gamma.

**TRP** Test Reference Profile.

**LFD** Latent Field Distribution.

**LCD** Latent Class Distribution.

**FieldMembership** Field membership probability matrix.

**ClassMembership** Class membership probability matrix.

**FieldEstimated** Estimated field assignment for each item.

**ClassEstimated** Estimated class assignment for each student.

**Students** Rank Membership Profile matrix with estimated class.

**FieldAnalysis** Field analysis with CRR and field memberships.

**TestFitIndices** Overall fit index for the test (binary layer, default).

**TestFitIndices\_nominal** Fit indices for the nominal layer (AIC/BIC/CAIC only).

**log\_lik** Log-likelihood of the binary layer model.

**log\_lik\_nominal** Log-likelihood of the nominal layer model.

**SOACflg** Logical; TRUE if Strongly Ordinal Alignment Condition is satisfied.

**WOACflg** Logical; TRUE if Weakly Ordinal Alignment Condition is satisfied.

## Examples

```
# Fit a Biclustering model with automatic structure learning using IRM
# gamma_c and gamma_f are concentration parameters for the Chinese Restaurant Process
result <- Biclustering_IRM(J35S515, gamma_c = 1, gamma_f = 1, verbose = TRUE)

# Display the Bicluster Reference Matrix (BRM) as a heatmap
plot(result, type = "Array")

# Plot Field Reference Profiles (FRP) in a 3-column grid
plot(result, type = "FRP", nc = 3)

result <- Biclustering_IRM(J35S515, gamma_c = 1, gamma_f = 1, verbose = TRUE)
plot(result, type = "Array")
```

```
# Fit a nominal Biclustering IRM model
result <- Biclustering_IRM(J20S600, gamma_c = 1, gamma_f = 1, verbose = TRUE)
plot(result, type = "Array")

# Fit an ordinal Biclustering IRM model
result <- Biclustering_IRM(J35S500, gamma_c = 1, gamma_f = 1, verbose = TRUE)
plot(result, type = "Array")

# Fit a rated Biclustering IRM model
result <- Biclustering_IRM(J21S300, gamma_c = 1, gamma_f = 1, verbose = TRUE)
plot(result, type = "Array")
```

---

 BINET

*Bicluster Network Model*


---

## Description

Bicluster Network Model: BINET is a model that combines the Bayesian network model and Biclustering. BINET is very similar to LDB and LDR. The most significant difference is that in LDB, the nodes represent the fields, whereas in BINET, they represent the class. BINET explores the local dependency structure among latent classes at each latent field, where each field is a locus.

## Usage

```
BINET(
  U,
  Z = NULL,
  w = NULL,
  na = NULL,
  conf = NULL,
  ncls = NULL,
  nfld = NULL,
  g_list = NULL,
  adj_list = NULL,
  adj_file = NULL,
  verbose = FALSE
)
```

## Arguments

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
Z	Z is a missing indicator matrix of the type matrix or data.frame
w	w is item weight vector
na	na argument specifies the numbers or characters to be treated as missing values.

<code>conf</code>	For the confirmatory parameter, you can input either a vector with items and corresponding fields in sequence, or a field membership profile matrix. In the case of the former, the field membership profile matrix will be generated internally. When providing a membership profile matrix, it needs to be either matrix or data.frame. The number of fields(nfld) will be overwrite to the number of columns of this matrix.
<code>ncls</code>	number of classes
<code>nfld</code>	number of fields
<code>g_list</code>	A list compiling graph-type objects for each rank/class.
<code>adj_list</code>	A list compiling matrix-type adjacency matrices for each rank/class.
<code>adj_file</code>	A file detailing the relationships of the graph for each rank/class, listed in the order of starting point, ending point, and rank(class).
<code>verbose</code>	verbose output Flag. default is TRUE

### Value

<b>nobs</b>	Sample size. The number of rows in the dataset.
<b>msg</b>	A character string indicating the model type.
<b>testlength</b>	Length of the test. The number of items included in the test.
<b>Nclass</b>	Optimal number of classes.
<b>Nfield</b>	Optimal number of fields.
<b>crr</b>	Correct Response Rate
<b>ItemLabel</b>	Label of Items
<b>FieldLabel</b>	Label of Fields
<b>all_adj</b>	Integrated Adjacency matrix used to plot graph.
<b>all_g</b>	Integrated graph object used to plot graph.see also <a href="#">plot.exametrika</a>
<b>adj_list</b>	List of Adjacency matrix used in the model
<b>params</b>	A list of the estimated conditional probabilities. It indicates which path was obtained from which parent node(class) to which child node(class), held by <code>parent</code> , <code>child</code> , and <code>field</code> . The item <code>Items</code> contained in the field is in <code>fld</code> . Named <code>chap</code> includes the conditional correct response answer rate of the child node, while <code>pap</code> contains the pass rate of the parent node.
<b>PSRP</b>	Response pattern by the students belonging to the parent classes of Class <code>c</code> . A more comprehensible arrangement of <code>params</code> .
<b>LCD</b>	Latent Class Distribution. see also <a href="#">plot.exametrika</a>
<b>LFD</b>	Latent Field Distribution. see also <a href="#">plot.exametrika</a>
<b>CMD</b>	Class Membership Distribution.
<b>FRP</b>	Marginal bicluster reference matrix.
<b>FRPIndex</b>	Index of FFP includes the item location parameters <code>B</code> and <code>Beta</code> , the slope parameters <code>A</code> and <code>Alpha</code> , and the monotonicity indices <code>C</code> and <code>Gamma</code> .
<b>TRP</b>	Test Reference Profile

**LDPSR** A rearranged set of parameters for output. It includes the field the items contained within that field, and the conditional correct response rate of parent nodes(class) and child node(class).

**FieldEstimated** Given vector which correspondence between items and the fields.

**Students** Rank Membership Profile matrix. The  $s$ -th row vector of  $\hat{M}_R, \hat{m}_R$ , is the rank membership profile of Student  $s$ , namely the posterior probability distribution representing the student's belonging to the respective latent classes.

**NextStage** The next class that easiest for students to move to, its membership probability, class-up odds, and the field required for more.

**MG\_FitIndices** Multigroup as Null model. See also [TestFit](#)

**SM\_FitIndices** Saturated Model as Null model. See also [TestFit](#)

## Examples

```
# Example: Bicluster Network Model (BINET)
# BINET combines Bayesian network model and Biclustering to explore
# local dependency structure among latent classes at each field

# Create field configuration vector based on field assignments
conf <- c(
  1, 5, 5, 5, 9, 9, 6, 6, 6, 6, 2, 7, 7, 11, 11, 7, 7,
  12, 12, 12, 2, 2, 3, 3, 4, 4, 4, 8, 8, 12, 1, 1, 6, 10, 10
)

# Create edge data for network structure between classes
edges_data <- data.frame(
  "From Class (Parent) >>>" = c(
    1, 2, 3, 4, 5, 7, 2, 4, 6, 8, 10, 6, 6, 11, 8, 9, 12
  ),
  ">>> To Class (Child)" = c(
    2, 4, 5, 5, 6, 11, 3, 7, 9, 12, 12, 10, 8, 12, 12, 11, 13
  ),
  "At Field (Locus)" = c(
    1, 2, 2, 3, 4, 4, 5, 5, 5, 5, 5, 7, 8, 8, 9, 9, 12
  )
)

# Save edge data to temporary CSV file
tmp_file <- tempfile(fileext = ".csv")
write.csv(edges_data, file = tmp_file, row.names = FALSE)

# Fit Bicluster Network Model
result.BINET <- BINET(
  J35S515,
  ncls = 13, # Maximum class number from edges (13)
  nflld = 12, # Maximum field number from conf (12)
  conf = conf, # Field configuration vector
  adj_file = tmp_file # Path to the CSV file
)

# Clean up temporary file
```

```
unlink(tmp_file)

# Display model results
print(result.BINET)

# Visualize different aspects of the model
plot(result.BINET, type = "Array") # Show bicluster structure
plot(result.BINET, type = "TRP") # Test Response Profile
plot(result.BINET, type = "LRD") # Latent Rank Distribution
plot(result.BINET,
      type = "RMP", # Rank Membership Profiles
      students = 1:9, nc = 3, nr = 3
)
plot(result.BINET,
      type = "FRP", # Field Reference Profiles
      nc = 3, nr = 2
)
plot(result.BINET,
      type = "LDPSR", # Locally Dependent Passing Student Rates
      nc = 3, nr = 2
)
```

---

BiserialCorrelation    *Biserial Correlation*

---

### Description

A biserial correlation is a correlation between dichotomous-ordinal and continuous variables.

### Usage

```
BiserialCorrelation(i, t)
```

### Arguments

i	i is a dichotomous-ordinal variable (0/1). x and y can also be the other way around.
t	t is a continuous variable. x and y can also be the other way around.

### Value

The biserial correlation coefficient between the two variables.

---

BitRespPtn	<i>Binary pattern maker</i>
------------	-----------------------------

---

**Description**

Binary pattern maker

**Usage**

```
BitRespPtn(n)
```

**Arguments**

n                    decimal numbers

**Details**

if n <- 1, return 0,1 if n <- 2, return 00,01,10,11 and so on.

**Value**

binary patterns

---

BNM	<i>Bayesian Network Model</i>
-----	-------------------------------

---

**Description**

performs Bayesian Network Model with specified graph structure

**Usage**

```
BNM(  
  U,  
  Z = NULL,  
  w = NULL,  
  na = NULL,  
  g = NULL,  
  adj_file = NULL,  
  adj_matrix = NULL,  
  beta1 = 1,  
  beta2 = 1  
)
```

**Arguments**

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
Z	Z is a missing indicator matrix of the type matrix or data.frame
w	w is item weight vector
na	na argument specifies the numbers or characters to be treated as missing values.
g	Specify a graph object suitable for the igraph class.
adj_file	specify CSV file where the graph structure is specified.
adj_matrix	specify adjacency matrix.
beta1	Beta distribution parameter 1 (for correct responses). Default is 1.
beta2	Beta distribution parameter 2 (for incorrect responses). Default is 1. Note: referred to as beta0 internally.

**Details**

This function performs a Bayesian network analysis on the relationships between items. This corresponds to Chapter 8 of the text. It uses the igraph package for graph visualization and checking the adjacency matrix. You need to provide either a graph object or a CSV file where the graph structure is specified.

**Value**

**nobs** Sample size. The number of rows in the dataset.  
**testlength** Length of the test. The number of items included in the test.  
**crr** correct response ratio  
**TestFitIndices** Overall fit index for the test. See also [TestFit](#)  
**adj** Adjacency matrix\  
**param** Learned Parameters  
**CCRR\_table** Correct Response Rate tables

**Examples**

```
# Create a Directed Acyclic Graph (DAG) structure for item relationships
# Each row represents a directed edge from one item to another
DAG <-
  matrix(
    c(
      "Item01", "Item02", # Item01 influences Item02
      "Item02", "Item03", # Item02 influences Item03
      "Item02", "Item04", # Item02 influences Item04
      "Item03", "Item05", # Item03 influences Item05
      "Item04", "Item05" # Item04 influences Item05
    ),
    ncol = 2, byrow = TRUE
  )
```

```
# Convert the DAG matrix to an igraph object for network analysis
g <- igraph::graph_from_data_frame(DAG)
g

# Create adjacency matrix from the graph
# Shows direct connections between items (1 for connection, 0 for no connection)
adj_mat <- as.matrix(igraph::as_adjacency_matrix(g))
print(adj_mat)

# Fit Bayesian Network Model using the specified adjacency matrix
# Analyzes probabilistic relationships between items based on the graph structure
result.BNM <- BNM(J5S10, adj_matrix = adj_mat)
result.BNM
```

---

BNM\_GA

*Structure Learning for BNM by simple GA*

---

## Description

Generating a DAG from data using a genetic algorithm.

## Usage

```
BNM_GA(  
  U,  
  Z = NULL,  
  w = NULL,  
  na = NULL,  
  seed = 123,  
  population = 20,  
  Rs = 0.5,  
  Rm = 0.005,  
  maxParents = 2,  
  maxGeneration = 100,  
  successiveLimit = 5,  
  crossover = 0,  
  elitism = 0,  
  filename = NULL,  
  verbose = TRUE  
)
```

## Arguments

U                    U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the [dataFormat](#) function.

Z	Z is a missing indicator matrix of the type matrix or data.frame
w	w is item weight vector
na	na argument specifies the numbers or characters to be treated as missing values.
seed	seed for random.
population	Population size. The default is 20
Rs	Survival Rate. The default is 0.5
Rm	Mutation Rate. The default is 0.005
maxParents	Maximum number of edges emanating from a single node. The default is 2.
maxGeneration	Maximum number of generations.
successiveLimit	Termination conditions. If the optimal individual does not change for this number of generations, it is considered to have converged.
crossover	Configure crossover using numerical values. Specify 0 for uniform crossover, where bits are randomly copied from both parents. Choose 1 for single-point crossover with one crossover point, and 2 for two-point crossover with two crossover points. The default is 0.
elitism	Number of elites that remain without crossover when transitioning to the next generation.
filename	Specify the filename when saving the generated adjacency matrix in CSV format. The default is null, and no output is written to the file.
verbose	verbose output Flag. default is TRUE

### Details

This function generates a DAG from data using a genetic algorithm. Depending on the size of the data and the settings, the computation may take a significant amount of computational time. For details on the settings or algorithm, see Shojima(2022), section 8.5

### Value

**adj** Optimal adjacency matrix

**testlength** Length of the test. The number of items included in the test.

**TestFitIndices** Overall fit index for the test. See also [TestFit](#)

**nobs** Sample size. The number of rows in the dataset.

**testlength** Length of the test. The number of items included in the test.

**crr** correct response ratio

**TestFitIndices** Overall fit index for the test. See also [TestFit](#)

**adj** Adjacency matrix

**param** Learned Parameters

**CCRR\_table** Correct Response Rate tables

**Examples**

```
# Perform Structure Learning for Bayesian Network Model using Genetic Algorithm
# Parameters are set for balanced exploration and computational efficiency
BNM_GA(J5S10,
  population = 20, # Size of population in each generation
  Rs = 0.5, # 50% survival rate for next generation
  Rm = 0.002, # 0.2% mutation rate for genetic diversity
  maxParents = 2, # Maximum of 2 parent nodes per item
  maxGeneration = 100, # Maximum number of evolutionary steps
  crossover = 2, # Use two-point crossover method
  elitism = 2 # Keep 2 best solutions in each generation
)
```

---

 BNM\_PBIL

*Structure Learning for BNM by PBIL*


---

**Description**

Generating a DAG from data using a Population-Based Incremental Learning

**Usage**

```
BNM_PBIL(
  U,
  Z = NULL,
  w = NULL,
  na = NULL,
  seed = 123,
  population = 20,
  Rs = 0.5,
  Rm = 0.002,
  maxParents = 2,
  maxGeneration = 100,
  successiveLimit = 5,
  elitism = 0,
  alpha = 0.05,
  estimate = 1,
  filename = NULL,
  verbose = TRUE
)
```

**Arguments**

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
Z	Z is a missing indicator matrix of the type matrix or data.frame

w	w is item weight vector
na	na argument specifies the numbers or characters to be treated as missing values.
seed	seed for random.
population	Population size. The default is 20
Rs	Survival Rate. The default is 0.5
Rm	Mutation Rate. The default is 0.002
maxParents	Maximum number of edges emanating from a single node. The default is 2.
maxGeneration	Maximum number of generations.
successiveLimit	Termination conditions. If the optimal individual does not change for this number of generations, it is considered to have converged.
elitism	Number of elites that remain without crossover when transitioning to the next generation.
alpha	Learning rate. The default is 0.05
estimate	In PBIL for estimating the adjacency matrix, specify by number from the following four methods: 1. Optimal adjacency matrix, 2. Rounded average of individuals in the last generation, 3. Rounded average of survivors in the last generation, 4. Rounded generational gene of the last generation. The default is 1.
filename	Specify the filename when saving the generated adjacency matrix in CSV format. The default is null, and no output is written to the file.
verbose	verbose output Flag. default is TRUE

### Details

This function performs structural learning using the Population-Based Incremental Learning model(PBIL) proposed by Fukuda et al.(2014) within the genetic algorithm framework. Instead of learning the adjacency matrix itself, the 'genes of genes' that generate the adjacency matrix are updated with each generation. For more details, please refer to Fukuda(2014) and Section 8.5.2 of the text(Shojima,2022).

### Value

**adj** Optimal adjacency matrix  
**testlength** Length of the test. The number of items included in the test.  
**TestFitIndices** Overall fit index for the test. See also [TestFit](#)  
**nobs** Sample size. The number of rows in the dataset.  
**testlength** Length of the test. The number of items included in the test.  
**crr** correct response ratio  
**TestFitIndices** Overall fit index for the test. See also [TestFit](#)  
**param** Learned Parameters  
**CCRR\_table** Correct Response Rate tables

## References

Fukuda, S., Yamanaka, Y., & Yoshihiro, T. (2014). A Probability-based evolutionary algorithm with mutations to learn Bayesian networks. *International Journal of Artificial Intelligence and Interactive Multimedia*, 3, 7–13. DOI: 10.9781/ijimai.2014.311

## Examples

```
# Perform Structure Learning for Bayesian Network Model using PBIL
# (Population-Based Incremental Learning)
BNM_PBIL(J5S10,
  population = 20, # Size of population in each generation
  Rs = 0.5, # 50% survival rate for next generation
  Rm = 0.005, # 0.5% mutation rate for genetic diversity
  maxParents = 2, # Maximum of 2 parent nodes per item
  alpha = 0.05, # Learning rate for probability update
  estimate = 4 # Use rounded generational gene method
)
```

---

calcFitIndices

*calc Fit Indices*

---

## Description

A general function that returns the model fit indices.

## Usage

```
calcFitIndices(chi_A, chi_B, df_A, df_B, nobs)
```

## Arguments

chi_A	chi-squares for this model
chi_B	chi-squares for compared model
df_A	degrees of freedom for this model
df_B	degrees of freedom for compared model
nobs	number of observations for Information criteria

## Value

**NFI** Normed Fit Index. Lager values closer to 1.0 indicate a better fit.

**RFI** Relative Fit Index. Lager values closer to 1.0 indicate a better fit.

**IFI** Incremental Fit Index. Lager values closer to 1.0 indicate a better fit.

**TLI** Tucker-Lewis Index. Lager values closer to 1.0 indicate a better fit.

**CFI** Comparative Fit Index. Lager values closer to 1.0 indicate a better fit.

**RMSEA** Root Mean Square Error of Approximation. Smaller values closer to 0.0 indicate a better fit.

**AIC** Akaike Information Criterion. A lower value indicates a better fit.

**CAIC** Consistent AIC. A lower value indicates a better fit.

**BIC** Bayesian Information Criterion. A lower value indicates a better fit.

CCRR

*Conditional Correct Response Rate*

### Description

The conditional correct response rate (CCRR) represents the ratio of the students who passed Item C (consequent item) to those who passed Item A (antecedent item). This function is applicable only to binary response data.

### Usage

```
CCRR(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
CCRR(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
CCRR(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'nominal'
```

```
CCRR(U, na = NULL, Z = NULL, w = NULL)
```

### Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

### Value

A matrix of conditional correct response rates with exametrika class. Each element (i,j) represents the probability of correctly answering item j given that item i was answered correctly.

### Note

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

**Examples**

```
# example code
# Calculate CCRR using sample dataset J5S10
CCRR(J5S10)
```

---

crr *Correct Response Rate*

---

**Description**

The correct response rate (CRR) is one of the most basic and important statistics for item analysis. This is an index of item difficulty and a measure of how many students out of those who tried an item correctly responded to it. This function is applicable only to binary response data.

The CRR for each item is calculated as:

$$p_j = \frac{\sum_{i=1}^n z_{ij} u_{ij}}{\sum_{i=1}^n z_{ij}}$$

where  $z_{ij}$  is the missing indicator and  $u_{ij}$  is the response.

**Usage**

```
crr(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
crr(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
crr(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

A numeric vector of weighted correct response rates for each item. Values range from 0 to 1, where higher values indicate easier items (more students answered correctly).

**Note**

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

**Examples**

```
# using sample dataset
crr(J15S500)
```

---

 CSR

*Conditional Selection Rate*


---

**Description**

Calculate the Conditional Selection Rate (CSR) for polytomous data. CSR measures the proportion of respondents who selected a specific category in item K, given that they selected a particular category in item J.

**Usage**

```
CSR(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Details**

The function returns a nested list structure CSR, where `CSR[[j]][[k]]` contains a matrix of conditional probabilities. In this matrix, the element at row *l* and column *m* represents  $P(K=m|J=l)$ , which is the probability of selecting category *m* for item K, given that category *l* was selected for item J.

Mathematically, for each cell (*l,m*) in the `CSR[[j]][[k]]` matrix:  $CSR[[j]][[k]][l,m] = P(\text{Item } K = \text{category } m \mid \text{Item } J = \text{category } l)$

This is calculated as the number of respondents who selected both category *l* for item J and category *m* for item K, divided by the total number of respondents who selected category *l* for item J.

**Value**

A list of Joint Selection Rate matrices for each item pair.

## Examples

```
# example code
# Calculate CSR using sample dataset J5S1000
CSR(J5S1000)

# Extract the conditional selection rates from item 1 to item 2
csr_1_2 <- CSR(J5S1000)[[1]][[2]]
# This shows the probability of selecting each category in item 2
# given that a specific category was selected in item 1
```

---

 CTT

*Classical Test Theory*


---

## Description

This function calculates the overall alpha and omega coefficients for the given data matrix. It also computes the alpha coefficient for each item, assuming that item is excluded.

## Usage

```
CTT(U, na = NULL, Z = NULL, w = NULL)
```

## Arguments

U	U is a data matrix of the type matrix or data.frame.
na	na argument specifies the numbers or characters to be treated as missing values.
Z	Z is a missing indicator matrix of the type matrix or data.frame
w	w is item weight vector

## Value

Returns a list of class c("exametrika", "CTT") containing two data frames:

**Reliability** A data frame with overall reliability coefficients (Alpha and Omega) calculated using different correlation matrices (Covariance, Phi, and Tetrachoric)

**ReliabilityExcludingItem** A data frame showing alpha coefficients when each item is excluded, calculated using different correlation matrices

## Examples

```
# using sample dataset
CTT(J15S500)
```

---

dataFormat	<i>dataFormat</i>
------------	-------------------

---

### Description

This function serves the role of formatting the data prior to the analysis.

### Usage

```
dataFormat(
  data,
  na = NULL,
  id = NULL,
  Z = NULL,
  w = NULL,
  response.type = NULL,
  CA = NULL
)
```

### Arguments

<code>data</code>	is a data matrix of the type <code>matrix</code> or <code>data.frame</code> .
<code>na</code>	<code>na</code> argument specifies the numbers or characters to be treated as missing values.
<code>id</code>	<code>id</code> indicates the column number containing the examinee ID. If <code>NULL</code> (default), the first column is auto-detected as ID or response data. If a column number is specified, that column is always used as the ID column.
<code>Z</code>	<code>Z</code> is a missing indicator matrix of the type <code>matrix</code> or <code>data.frame</code>
<code>w</code>	<code>w</code> is item weight vector
<code>response.type</code>	Character string specifying the type of response data: "binary" for dichotomous data, "ordinal" for ordered polytomous data, "rated" for polytomous data with correct answers, "nominal" for unordered polytomous data. If <code>NULL</code> (default), the type is automatically detected.
<code>CA</code>	A numeric vector specifying the correct answers for rated polytomous data. Required when <code>response.type</code> is "rated".

### Value

**U** For binary response data. A matrix with rows representing the sample size and columns representing the number of items, where elements are either 0 or 1.  $u_{ij} = 1$  indicates that student  $i$  correctly answered item  $j$ , while  $u_{ij} = 0$  means that student  $i$  answered item  $j$  incorrectly.

**Q** For polytomous response data. A matrix with rows representing the sample size and columns representing the number of items, where elements are non-negative integers. When input data is in factor format, the factor levels are converted to consecutive integers starting from 1.

**ID** The ID label given by the designated column or function.

**ItemLabel** The item names given by the provided column names or function.

**Z** Missing indicator matrix.  $z_{ij} = 1$  indicates that item  $j$  is presented to Student  $i$ , while  $z_{ij} = 0$  indicates item  $j$  is NOT presented to Student  $i$ . If the data contains NA values, -1 is assigned.

**w** Item weight vector

**response.type** Character string indicating the type of response data: "binary", "ordinal", "rated", or "nominal"

**CategoryLabel** List containing the original factor labels when polytomous responses are provided as factors. NULL if no factor data is present.

**categories** Numeric vector containing the number of response categories for each item.

**CA** For rated polytomous data, a numeric vector of correct answers. NULL for other types.

---

Dimensionality

*Dimensionality*

---

## Description

The dimensionality is the number of components the test is measuring.

## Usage

```
Dimensionality(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
Dimensionality(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
Dimensionality(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'rated'
```

```
Dimensionality(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'ordinal'
```

```
Dimensionality(U, na = NULL, Z = NULL, w = NULL)
```

## Arguments

- |    |   |
|----|---|
| U  | Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function. |
| na | Values to be treated as missing values.   |
| Z  | Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.                                   |
| w  | Item weight vector specifying the relative importance of each item.   |

**Value**

Returns a list of class c("exametrika", "Dimensionality") containing:

**Component** Sequence of component numbers

**Eigenvalue** Eigenvalues of the tetrachoric correlation matrix

**PerOfVar** Percentage of variance explained by each component

**CumOfPer** Cumulative percentage of variance explained

---

DistractorAnalysis      *Distractor Analysis*

---

**Description**

Performs distractor analysis for rated (multiple-choice) models. For each item and rank/class, computes observed category frequency tables, chi-square tests against chance level (uniform distribution), and Cramer's V as an effect size measure.

This function works with results from [LRA](#) (rated data) and [Biclustering / Biclustering\\_IRM](#) (rated data).

**Usage**

```
DistractorAnalysis(x, ...)

## S3 method for class 'LRArated'
DistractorAnalysis(x, ...)

## S3 method for class 'ratedBiclustering'
DistractorAnalysis(x, ...)

## S3 method for class 'DistractorAnalysis'
print(x, items = NULL, ranks = NULL, digits = 4, ...)

## S3 method for class 'DistractorAnalysis'
plot(x, type = "Distractor", items = NULL, ranks = NULL, nc = 1, nr = 1, ...)
```

**Arguments**

x	A result object from a rated model (class LRArated or ratedBiclustering).
...	Additional arguments (currently unused).
items	Integer vector of item indices to display. NULL for all items.
ranks	Integer vector of rank/class indices to display. NULL for all ranks.
digits	Number of digits for rounding. Default is 4.
type	Plot type. Currently only "Distractor" is supported.
nc	Number of columns in the plot grid.
nr	Number of rows in the plot grid.

**Value**

An object of class c("exametrika", "DistractorAnalysis") containing:

**freq\_table** List of frequency matrices (nrank x maxQ), one per item.

**prop\_table** List of proportion matrices (nrank x maxQ), one per item.

**chisq\_table** Matrix (nitems x nrank) of chi-square statistics.

**pvalue\_table** Matrix (nitems x nrank) of p-values.

**cramersv\_table** Matrix (nitems x nrank) of Cramer's V effect sizes.

**CA** Correct answer vector.

**n\_rank** Number of ranks/classes.

**maxQ** Number of response categories.

**nitems** Number of items.

**ItemLabel** Item label vector.

**n\_field** Number of fields (Biclustering only).

**FieldEstimated** Field assignment vector (Biclustering only).

**field\_items** List of item indices per field (Biclustering only).

**Examples**

```
# LRA.rated example
result_lra <- LRA(J21S300, nrank = 5, mic = TRUE)
da <- DistractorAnalysis(result_lra)
print(da)
print(da, items = 1:3)
print(da, ranks = c(1, 5))
plot(da)
plot(da, items = 1:6, nc = 3, nr = 2)

# Biclustering.rated example
result_bic <- Biclustering(J21S300, ncls = 5, nfld = 3, method = "R")
da_bic <- DistractorAnalysis(result_bic)
print(da_bic, items = 1:7)
plot(da_bic, items = 1:6, nc = 3, nr = 2)
```

**Description**

Performs a grid search to find optimal parameters for different analysis methods. Supports Biclustering, LCA (Latent Class Analysis), and LRA (Latent Rank Analysis).

**Usage**

```
GridSearch(
  obj,
  max_ncls = 10,
  max_nfld = 10,
  fun = "Biclustering",
  index = "BIC",
  verbose = TRUE,
  ...
)
```

**Arguments**

obj	Input data matrix or object to be analyzed
max_ncls	Maximum number of classes/clusters to test (default: 10)
max_nfld	Maximum number of fields to test for Biclustering (default: 10)
fun	Function name to use for analysis. Options: "Biclustering", "LCA", "LRA" (default: "Biclustering")
index	Fit index to optimize from TestFitIndices returned by each function. Valid options: "BIC" (default), "AIC", "CAIC", "model_log_like", "model_Chi_sq", "RMSEA", "NFI", "RFI", "IFI", "TLI", "CFI". Aliases are also accepted: "log_lik", "log_lik", "LogLik", "LL" (all map to "model_log_like"), "Chi_sq", "chi_sq" (map to "model_Chi_sq").
verbose	Logical; if TRUE, displays detailed progress messages during grid search. Default is TRUE.
...	Additional arguments passed to the analysis function

**Value**

A list containing: For Biclustering:

index_matrix	Matrix of fit indices for each ncls/nfld combination
optimal_ncls	Optimal number of classes/clusters
optimal_nfld	Optimal number of fields
optimal_result	Analysis result using optimal parameters
failed_settings	List of parameter combinations that failed to converge

For LCA/LRA:

index_vec	Vector of fit indices for each ncls
optimal_ncls	Optimal number of classes/clusters
optimal_result	Analysis result using optimal parameters
failed_settings	List of parameter combinations that failed to converge

## Examples

```
## Not run:
# Grid search for Biclustering
result <- grid_serch(data_matrix, max_ncls = 5, max_nfld = 5)

# Grid search for LCA
result <- grid_serch(data_matrix, max_ncls = 8, fun = "LCA")

## End(Not run)
```

---

 GRM

*Graded Response Model (GRM)*


---

## Description

Implements Samejima's (1969) Graded Response Model (GRM), which is an Item Response Theory model for ordered categorical response data. The model estimates discrimination parameters and category threshold parameters for each item. It is widely used in psychological measurement, educational assessment, and other fields that deal with multi-step rating scales.

## Usage

```
GRM(U, na = NULL, Z = NULL, w = NULL, verbose = TRUE)
```

## Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class using the <a href="#">dataFormat</a> function.
na	Specifies numbers or characters to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. 1 indicates observed values, 0 indicates missing values.
w	Item weight vector
verbose	Logical; if TRUE, shows progress of iterations (default: TRUE)

## Value

A list of class "exametrika" and "GRM" containing the following elements:

- testlength** Length of the test (number of items)
- nobs** Sample size (number of rows in the dataset)
- log\_lik** Log-likelihood value at convergence
- iterations** Number of iterations and function evaluations from optimization
- params** Matrix containing the estimated item parameters
- EAP** Ability parameters of examinees estimated by EAP method

**MAP** Ability parameters of examinees estimated by MAP method

**PSD** Posterior standard deviation of the ability parameters

**ItemFitIndices** Fit indices for each item. See also [ItemFit](#)

**TestFitIndices** Overall fit indices for the test. See also [TestFit](#)

## References

Samejima, F. (1969). Estimation of latent ability using a response pattern of graded scores. *Psychometrika Monograph Supplement*, 34(4, Pt. 2), 1-100.

## Examples

```
## Not run:
# Apply GRM to example data
result <- GRM(J5S1000)
print(result)
plot(result, type = "IRF")
plot(result, type = "IIF")
plot(result, type = "TIF")

## End(Not run)
```

---

grm\_iif

*Item Information Function for GRM*

---

## Description

Calculates the value of the Item Information Function for the Graded Response Model.

## Usage

```
grm_iif(theta, a, b)
```

## Arguments

theta	Latent trait value of the subject
a	Discrimination parameter of IRF
b	Vector of difficulty parameters (thresholds) of IRF

## Value

Value of the Item Information Function

**Examples**

```
## Not run:  
# Example for an item with 3 categories  
a <- 1.5  
b <- c(-1.0, 1.0)  
thetas <- seq(-3, 3, by = 0.1)  
info <- sapply(thetas, function(t) grm_iif(t, a, b))  
plot(thetas, info, type = "l", xlab = "Theta", ylab = "Information")  
  
## End(Not run)
```

---

grm\_prob

*Probability function for GRM*

---

**Description**

Calculates the probability of selecting each category given a latent trait value and item parameters.

**Usage**

```
grm_prob(theta, a, b)
```

**Arguments**

theta	Latent trait value of the subject
a	Discrimination parameter of IRF
b	Vector of difficulty parameters (thresholds) of IRF

**Value**

Vector of category selection probabilities

**Examples**

```
## Not run:  
# Example for an item with 3 categories  
a <- 1.5  
b <- c(-1.0, 1.0)  
theta <- 0  
grm_prob(theta, a, b)  
  
## End(Not run)
```

---

 IIF2PLM

*IIF for 2PLM*


---

**Description**

Item Information Function for 2PLM

**Usage**

IIF2PLM(a, b, theta)

**Arguments**

a	slope parameter
b	location parameter
theta	ability parameter

**Value**

Returns a numeric vector representing the item information at each ability level theta. The information is calculated as:  $I(\theta) = a^2 P(\theta)(1 - P(\theta))$

---

 IIF3PLM

*IIF for 3PLM*


---

**Description**

Item Information Function for 3PLM

**Usage**

IIF3PLM(a, b, c, theta)

**Arguments**

a	slope parameter
b	location parameter
c	lower asymptote parameter
theta	ability parameter

**Value**

Returns a numeric vector representing the item information at each ability level theta. The information is calculated as:  $I(\theta) = \frac{a^2(1-P(\theta))(P(\theta)-c)^2}{(1-c)^2P(\theta)}$

---

 InterItemAnalysis      *Inter-Item Analysis for Psychometric Data*


---

**Description**

Calculates various relationship metrics between pairs of items in test data. This analysis helps identify item interdependencies, content overlaps, and potential local dependence. For binary data, metrics include joint response rates, conditional probabilities, and several correlation measures. For ordinal/rated data, appropriate correlation measures are calculated.

The following metrics are calculated for binary data:

- JSS: Joint Sample Size - number of examinees responding to both items
- JCRR: Joint Correct Response Rate - proportion of examinees answering both items correctly
- CCRR: Conditional Correct Response Rate - probability of answering one item correctly given a correct response to another item
- IL: Item Lift - ratio of joint correct response rate to the product of marginal rates
- MI: Mutual Information - measure of mutual dependence between items
- Phi: Phi Coefficient - correlation coefficient for binary variables
- Tetrachoric: Tetrachoric Correlation - estimate of Pearson correlation for underlying continuous variables

For ordinal/rated data, the function calculates:

- JSS: Joint Sample Size
- JSR: Joint Selection Rate
- CSR: Conditional Selection Rate
- MI: Mutual Information
- Polychoric: Polychoric Correlation - extension of tetrachoric correlation for ordinal data

**Usage**

```
InterItemAnalysis(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

## Details

This function automatically detects the data type and applies appropriate analysis methods:

- For binary data: Calculates tetrachoric correlations and related statistics
- For ordinal/rated data: Calculates polychoric correlations and related statistics
- For nominal data: Returns an error (not supported)

Inter-item analysis is useful for:

- Identifying groups of highly related items
- Detecting local dependence between items
- Evaluating test dimensionality
- Informing item selection and test construction

## Value

For binary data, an object of class "exametrika" and "IIAnalysis" containing:

**JSS** Joint Sample Size matrix -  $N(i,j)$  shows number of examinees who responded to both items  $i$  and  $j$

**JCRR** Joint Correct Response Rate matrix -  $P(X_i=1, X_j=1)$  shows probability of correct responses to both items

**CCRR** Conditional Correct Response Rate matrix -  $P(X_i=1|X_j=1)$  shows probability of correct response to item  $i$  given correct response to item  $j$

**IL** Item Lift matrix -  $P(X_i=1, X_j=1)/(P(X_i=1)*P(X_j=1))$  measures association strength

**MI** Mutual Information matrix - measures information shared between items

**Phi** Phi Coefficient matrix - correlation coefficient between binary variables

**Tetrachoric** Tetrachoric Correlation matrix - correlation between underlying continuous variables

For ordinal/rated data, an object of class "exametrika" and "IIAnalysis.ordinal" containing:

**JSS** Joint Sample Size matrix

**JSR** Joint Selection Rate matrix - frequencies of joint category selections

**CSR** Conditional Selection Rate matrix - probabilities of response categories conditional on other items

**MI** Mutual Information matrix

**Polychoric** Polychoric Correlation matrix - correlations between underlying continuous variables

## See Also

[dataFormat](#) for data preparation, [CTT](#) for Classical Test Theory analysis

## Examples

```
# Basic usage with binary data
ii_analysis <- InterItemAnalysis(J15S500)

# View joint sample sizes
head(ii_analysis$JSS)

# View tetrachoric correlations
head(ii_analysis$Tetrachoric)

# Find pairs of items with high mutual information (potential local dependence)
high_MI <- which(ii_analysis$MI > 0.2 & upper.tri(ii_analysis$MI), arr.ind = TRUE)
if (nrow(high_MI) > 0) {
  print("Item pairs with high mutual information:")
  print(high_MI)
}

# Example with ordinal data
ordinal_analysis <- InterItemAnalysis(J15S3810)

# View polychoric correlations for ordinal data
head(ordinal_analysis$Polychoric)
```

---

IRM

*IRM (Deprecated)*

---

## Description

This function has been renamed to [Biclustering\\_IRM](#). Please use `Biclustering_IRM` instead.

## Usage

```
IRM(...)
```

## Arguments

... All arguments passed to [Biclustering\\_IRM](#)

**Description**

A function for estimating item parameters using the EM algorithm.

**Usage**

```
IRT(U, model = 2, na = NULL, Z = NULL, w = NULL, verbose = TRUE)
```

**Arguments**

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
model	This argument takes the number of item parameters to be estimated in the logistic model. It is limited to values 2, 3, or 4.
na	na argument specifies the numbers or characters to be treated as missing values.
Z	Z is a missing indicator matrix of the type matrix or data.frame
w	w is item weight vector
verbose	logical; if TRUE, shows progress of iterations (default: TRUE)

**Details**

Apply the 2, 3, and 4 parameter logistic models to estimate the item and subject populations. The 4PL model can be described as follows.

$$P(\theta, a_j, b_j, c_j, d_j) = c_j + \frac{d_j - c_j}{1 + \exp\{-a_j(\theta - b_j)\}}$$

$a_j, b_j, c_j,$  and  $d_j$  are parameters related to item  $j$ , and are parameters that adjust the logistic curve.  $a_j$  is called the slope parameter,  $b_j$  is the location,  $c_j$  is the lower asymptote, and  $d_j$  is the upper asymptote parameter. The model includes lower models, and among the 4PL models, the case where  $d = 1$  is the 3PL model, and among the 3PL models, the case where  $c = 0$  is the 2PL model.

**Value**

**model** number of item parameters you set.

**testlength** Length of the test. The number of items included in the test.

**nobs** Sample size. The number of rows in the dataset.

**params** Matrix containing the estimated item parameters

**Q3mat** Q3-matrix developed by Yen(1984)

**itemPSD** Posterior standard deviation of the item parameters

**ability** Estimated parameters of students ability

**ItemFitIndices** Fit index for each item. See also [ItemFit](#)

**TestFitIndices** Overall fit index for the test. See also [TestFit](#)

## References

Yen, W. M. (1984) Applied Psychological Measurement, 8, 125-145.

## Examples

```
# Fit a 3-parameter IRT model to the sample dataset
result.IRT <- IRT(J15S500, model = 3)

# Display the first few rows of estimated student abilities
head(result.IRT$ability)

# Plot Item Response Function (IRF) for items 1-6 in a 2x3 grid
plot(result.IRT, type = "IRF", items = 1:6, nc = 2, nr = 3)

# Plot Item Information Function (IIF) for items 1-6 in a 2x3 grid
plot(result.IRT, type = "IIF", items = 1:6, nc = 2, nr = 3)

# Plot the Test Information Function (TIF) for all items
plot(result.IRT, type = "TIF")
```

---

ITBiserial

*Item-Total Biserial Correlation*

---

## Description

The Item-Total Biserial Correlation computes the biserial correlation between each item and the total score. This function is applicable only to binary response data.

This correlation provides a measure of item discrimination, indicating how well each item distinguishes between high and low performing examinees.

## Usage

```
ITBiserial(U, na = NULL, Z = NULL, w = NULL)

## Default S3 method:
ITBiserial(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'binary'
ITBiserial(U, na = NULL, Z = NULL, w = NULL)
```

## Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

A numeric vector of item-total biserial correlations. Values range from -1 to 1, where:

- Values near 1: Strong positive discrimination
- Values near 0: No discrimination
- Negative values: Potential item problems

**Note**

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

The biserial correlation is generally preferred over the point-biserial correlation when the dichotomization is artificial (i.e., when the underlying trait is continuous).

**Examples**

```
# using sample dataset
ITBiserial(J15S500)
```

---

ItemEntropy

*Item Entropy*

---

**Description**

The item entropy is an indicator of the variability or randomness of the responses. This function is applicable only to binary response data.

The entropy value represents the uncertainty or information content of the response pattern for each item, measured in bits. Maximum entropy (1 bit) occurs when correct and incorrect responses are equally likely ( $p = 0.5$ ).

**Usage**

```
ItemEntropy(U, na = NULL, Z = NULL, w = NULL)

## Default S3 method:
ItemEntropy(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'binary'
ItemEntropy(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'ordinal'
ItemEntropy(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Details**

The item entropy is calculated as:

$$e_j = -p_j \log_2 p_j - (1 - p_j) \log_2 (1 - p_j)$$

where  $p_j$  is the correct response rate for item j.

The entropy value has the following properties:

- Maximum value of 1 bit when  $p = 0.5$  (most uncertainty)
- Minimum value of 0 bits when  $p = 0$  or  $1$  (no uncertainty)
- Higher values indicate more balanced response patterns
- Lower values indicate more predictable response patterns

**Value**

A numeric vector of entropy values for each item, measured in bits. Values range from 0 to 1, where:

- 1: maximum uncertainty ( $p = 0.5$ )
- 0: complete certainty ( $p = 0$  or  $1$ )
- Values near 1 indicate items with balanced response patterns
- Values near 0 indicate items with extreme response patterns

**Note**

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

**Examples**

```
# using sample dataset
ItemEntropy(J5S10)
```

---

ItemFit

*Model Fit Functions for Items*


---

### Description

A general function that returns the model fit indices.

### Usage

```
ItemFit(U, Z, ell_A, nparam)
```

### Arguments

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
Z	Z is a missing indicator matrix of the type matrix or data.frame
ell_A	log likelihood of this model
nparam	number of parameters for this model

### Value

**model\_log\_like** log likelihood of analysis model

**bench\_log\_like** log likelihood of benchmark model

**null\_log\_like** log likelihood of null model

**model\_Chi\_sq** Chi-Square statistics for analysis model

**null\_Chi\_sq** Chi-Square statistics for null model

**model\_df** degrees of freedom of analysis model

**null\_df** degrees of freedom of null model

**NFI** Normed Fit Index. Lager values closer to 1.0 indicate a better fit.

**RFI** Relative Fit Index. Lager values closer to 1.0 indicate a better fit.

**IFI** Incremental Fit Index. Lager values closer to 1.0 indicate a better fit.

**TLI** Tucker-Lewis Index. Lager values closer to 1.0 indicate a better fit.

**CFI** Comparative Fit Index. Lager values closer to 1.0 indicate a better fit.

**RMSEA** Root Mean Square Error of Approximation. Smaller values closer to 0.0 indicate a better fit.

**AIC** Akaike Information Criterion. A lower value indicates a better fit.

**CAIC** Consistent AIC. A lower value indicates a better fit.

**BIC** Bayesian Information Criterion. A lower value indicates a better fit.

---

ItemInformationFunc *IIF for 4PLM*

---

### Description

Item Information Function for 4PLM

### Usage

```
ItemInformationFunc(a = 1, b, c = 0, d = 1, theta)
```

### Arguments

a	slope parameter
b	location parameter
c	lower asymptote parameter
d	upper asymptote parameter
theta	ability parameter

### Value

Returns a numeric vector representing the item information at each ability level theta. The information is calculated based on the first derivative of the log-likelihood of the 4PL model with respect to theta.

---

ItemLift *Item Lift*

---

### Description

The lift is a commonly used index in a POS data analysis. The item lift of Item k to Item j is defined as follow:  $l_{jk} = \frac{p_{k|j}}{p_k}$  This function is applicable only to binary response data.

### Usage

```
ItemLift(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
ItemLift(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
ItemLift(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

A matrix of item lift values with exametrika class. Each element (j,k) represents the lift value of item k given item j, which indicates how much more likely item k is to be correct given that item j was answered correctly.

**Note**

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

**References**

Brin, S., Motwani, R., Ullman, J., & Tsur, S. (1997). Dynamic itemset counting and implication rules for market basket data. In Proceedings of ACM SIGMOD International Conference on Management of Data (pp. 255–264). <https://dl.acm.org/doi/10.1145/253262.253325>

**Examples**

```
# example code
# Calculate ItemLift using sample dataset J5S10
ItemLift(J5S10)
```

---

ItemOdds

*Item Odds*


---

**Description**

Item Odds are defined as the ratio of Correct Response Rate to Incorrect Response Rate:

$$O_j = \frac{p_j}{1 - p_j}$$

where  $p_j$  is the correct response rate for item j. This function is applicable only to binary response data.

The odds value represents how many times more likely a correct response is compared to an incorrect response. For example, an odds of 2 means students are twice as likely to answer correctly as incorrectly.

**Usage**

```
ItemOdds(U, na = NULL, Z = NULL, w = NULL)

## Default S3 method:
ItemOdds(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'binary'
ItemOdds(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

A numeric vector of odds values for each item. Values range from 0 to infinity, where:

- odds > 1: correct response more likely than incorrect
- odds = 1: equally likely
- odds < 1: incorrect response more likely than correct

**Note**

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

**Examples**

```
# using sample dataset
ItemOdds(J5S10)
```

---

ItemReport

---

*Generate Item Report for Non-Binary Test Data*


---

**Description**

Calculates item-level statistics for non-binary test data, including response rates, basic descriptive statistics, and item-total correlations.

**Usage**

```
ItemReport(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item

**Details**

This function is intended for non-binary (ordinal or rated) response data. It provides detailed statistics for each item in the test, focusing on response patterns and the relationship between individual items and overall test performance. If binary data is provided, an error message will be displayed.

**Value**

An object of class "exametrika" and "QitemStatistics" containing:

**ItemLabel** Labels identifying each item

**Obs** Number of valid responses for each item

**ObsRatio** Proportion of valid responses for each item (range: 0-1)

**ItemMean** Mean score of each item

**ItemSD** Standard deviation of each item score

**ItemCORR** Item-total correlation coefficients - correlation between item scores and total test scores

**ItemCORR\_R** Corrected item-total correlation coefficients - correlation between item scores and total test scores excluding the target item

**Examples**

```
# Generate item report for sample ordinal data
item_stats <- ItemReport(J15S3810)

# View first few rows of the item report
head(item_stats)

# Example with rated data including custom missing value indicator
item_stats2 <- ItemReport(J35S5000, na = -99)
```

---

ItemStatistics	<i>Simple Item Statistics</i>
----------------	-------------------------------

---

**Description**

This function calculates statistics for each item, with different metrics available depending on the data type (binary, ordinal, or rated).

**Usage**

```
ItemStatistics(U, na = NULL, Z = NULL, w = NULL)

## Default S3 method:
ItemStatistics(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'binary'
ItemStatistics(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'ordinal'
ItemStatistics(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

For binary data:

**ItemLabel** Label identifying each item

**NR** Number of Respondents for each item

**CRR** Correct Response Rate denoted as  $p_j$ .

**ODDs** Item Odds is the ratio of the correct response rate to the incorrect response rate. Defined as

$$o_j = \frac{p_j}{1-p_j}$$

**Threshold** Item Threshold is a measure of difficulty based on a standard normal distribution.

**Entropy** Item Entropy is an indicator of the variability or randomness of the responses. Defined as

$$e_j = -p_j \log_2 p_j - (1 - p_j) \log_2 (1 - p_j)$$

**ITCrr** Item-total Correlation is a Pearson's correlation of an item with the Number-Right score.

For ordinal polytomous data:

**ItemLabel** Label identifying each item

**NR** Number of Respondents for each item

**Threshold** Matrix of threshold values for each item's category boundaries, based on a standard normal distribution. For an item with  $K$  categories, there are  $K-1$  thresholds.

**Entropy** Item Entropy calculated using the category probabilities. Unlike binary data, this is calculated using the formula  $e_j = -\sum_{k=1}^{K_j} p_{jk} \log_{K_j} p_{jk}$ , where  $K_j$  is the number of categories for item  $j$ .

**ITCrr** Item-total Correlation calculated using polyserial correlation, which accounts for the ordinal nature of the item responses and the continuous total score.

### Note

For rated data, the function processes the data as binary, with each response being compared to the correct answer to determine correctness.

### Examples

```
# using sample dataset(binary)
ItemStatistics(J15S500)
```

---

ItemThreshold	<i>Item Threshold</i>
---------------	-----------------------

---

### Description

Item threshold is a measure of difficulty based on a standard normal distribution. This function is applicable only to binary response data.

The threshold is calculated as:

$$\tau_j = \Phi^{-1}(1 - p_j)$$

where  $\Phi^{-1}$  is the inverse standard normal distribution function and  $p_j$  is the correct response rate for item  $j$ .

Higher threshold values indicate more difficult items, as they represent the point on the standard normal scale above which examinees tend to answer incorrectly.

### Usage

```
ItemThreshold(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
ItemThreshold(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'ordinal'
ItemThreshold(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

A numeric vector of threshold values for each item on the standard normal scale. Typical values range from about -3 to 3, where:

- Positive values indicate difficult items
- Zero indicates items of medium difficulty (50% correct)
- Negative values indicate easy items

**Note**

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

**Examples**

```
# using sample dataset
ItemThreshold(J5S10)
```

---

ItemTotalCorr	<i>Item-Total Correlation</i>
---------------	-------------------------------

---

**Description**

Item-Total correlation (ITC) is a Pearson's correlation of an item with the Number-Right Score (NRS) or total score. This function is applicable only to binary response data.

The ITC is a measure of item discrimination, indicating how well an item distinguishes between high and low performing examinees.

**Usage**

```
ItemTotalCorr(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
ItemTotalCorr(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
ItemTotalCorr(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'ordinal'  
ItemTotalCorr(U, na = NULL, Z = NULL, w = NULL)
```

### Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

### Details

The correlation is calculated between:

- Each item's responses (0 or 1)
- The total test score (sum of correct responses)

Higher positive correlations indicate items that better discriminate between high and low ability examinees.

### Value

A numeric vector of item-total correlations. Values typically range from -1 to 1, where:

- Values near 1: Strong positive discrimination
- Values near 0: No discrimination
- Negative values: Potential item problems (lower ability students performing better than higher ability students)

### Note

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

Values below 0.2 might indicate problematic items that should be reviewed. Values above 0.3 are generally considered acceptable.

### Examples

```
# using sample dataset  
ItemTotalCorr(J15S500)
```

---

J12S5000

*J12S5000*

---

**Description**

A binary response dataset for test analysis

**Usage**

J12S5000

**Format**

An exametrika class object with 5000 students and 12 items containing binary (0/1) responses

**Source**

<http://sh0j1ma.stars.ne.jp/exmk/>

---

J15S3810

*J15S3810*

---

**Description**

An ordinal response dataset for test analysis

**Usage**

J15S3810

**Format**

An exametrika class object with 3810 students and 15 items containing ordinal responses with 4 categories

---

J15S500

*J15S500*

---

**Description**

A binary response dataset for test analysis

**Usage**

J15S500

**Format**

An exametrika class object with 500 students and 15 items containing binary (0/1) responses

**Source**

<http://sh0j1ma.stars.ne.jp/exmk/>

---

J20S400

*J20S400*

---

**Description**

A binary response dataset for test analysis

**Usage**

J20S400

**Format**

An exametrika class object with 400 students and 20 items containing binary (0/1) responses

**Source**

<http://sh0j1ma.stars.ne.jp/exmk/>

---

*J20S600**J20S600*

---

**Description**

A simulated nominal dataset for polytomous Biclustering analysis. This is a synthetic dataset generated using random number generation with a cyclic category preference pattern (5 latent classes, 4 fields, 4 response categories) for demonstration and testing purposes. Contains approximately 0.5\

**Usage**

*J20S600*

**Format**

An exametrika class object with 600 students and 20 items containing nominal responses with 4 categories

---

*J21S300**J21S300*

---

**Description**

A simulated rated dataset for Rated IRM and Biclustering analysis. This is a synthetic dataset generated with a known structure (5 latent classes, 3 fields, 4 response categories with correct answers) for demonstration and testing purposes. Each field contains 7 items.

**Usage**

*J21S300*

**Format**

An exametrika class object with 300 students and 21 items containing rated responses (multiple-choice with correct answers) with 4 categories

---

*J35S500**J35S500*

---

**Description**

A simulated ordinal dataset for polytomous Biclustering analysis. This is a synthetic dataset generated using random number generation with a cumulative staircase pattern (5 latent classes, 5 fields, 5 response categories) for demonstration and testing purposes. Contains approximately 0.5\

**Usage**

J35S500

**Format**

An exametrika class object with 500 students and 35 items containing ordinal responses with 5 categories

---

*J35S5000**J35S5000*

---

**Description**

A rated response dataset for test analysis

**Usage**

J35S5000

**Format**

An exametrika class object with 5000 students and 35 items containing polytomous responses with correct answers

---

J35S515

*J35S515*

---

**Description**

A binary response dataset for test analysis

**Usage**

J35S515

**Format**

An exametrika class object with 515 students and 35 items containing binary (0/1) responses

**Source**

<http://sh0j1ma.stars.ne.jp/exmk/>

---

J50S100

*J50S100*

---

**Description**

A simulated binary dataset for test analysis. This is a synthetic dataset generated using random number generation for demonstration and testing purposes.

**Usage**

J50S100

**Format**

An exametrika class object with 100 students and 50 items containing binary responses

---

J5S10

*J5S10*

---

**Description**

A binary response dataset for test analysis

**Usage**

J5S10

**Format**

An exametrika class object with 5 students and 10 items containing binary (0/1) responses

**Source**

<http://sh0j1ma.stars.ne.jp/exmk/>

---

J5S1000

*J5S1000*

---

**Description**

A simulated ordinal dataset for test analysis. This is a synthetic dataset generated using random number generation for demonstration and testing purposes.

**Usage**

J5S1000

**Format**

An exametrika class object with 1000 students and 5 items containing ordinal responses

---

JCRR

*Joint Correct Response Rate*

---

### Description

The joint correct response rate (JCRR) is the rate of students who passed both items. This function is applicable only to binary response data. For non-binary data, it will automatically redirect to the JSR function with an appropriate message.

### Usage

```
JCRR(U, na = NULL, Z = NULL, w = NULL)

## Default S3 method:
JCRR(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'binary'
JCRR(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'nominal'
JCRR(U, na = NULL, Z = NULL, w = NULL)
```

### Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

### Value

A matrix of joint correct response rates with exametrika class. Each element (i,j) represents the proportion of students who correctly answered both items i and j.

### Note

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

### Examples

```
# example code
# Calculate JCRR using sample dataset J5S10
JCRR(J5S10)
```

---

JointSampleSize	<i>Joint Sample Size</i>
-----------------	--------------------------

---

### Description

The joint sample size is a matrix whose elements are the number of individuals who responded to each pair of items.

### Usage

```
JointSampleSize(U, na = NULL, Z = NULL, w = NULL)

## Default S3 method:
JointSampleSize(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'binary'
JointSampleSize(U, na = NULL, Z = NULL, w = NULL)
```

### Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

### Value

Returns a matrix of class `c("exametrika", "matrix")` where each element  $(i,j)$  represents the number of students who responded to both item  $i$  and item  $j$ . The diagonal elements represent the total number of responses for each item.

### Note

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

---

JSR

*Joint Selection Rate*

---

### Description

Calculate the Joint Selection Rate (JSR) for polytomous data. JSR measures the proportion of respondents who selected specific category combinations between pairs of items. For each pair of items (j,k), it returns a contingency table showing the joint probability of selecting each category combination.

### Usage

```
JSR(U, na = NULL, Z = NULL, w = NULL)
```

### Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

### Value

A list of Joint Selection Rate matrices for each item pair.

### Examples

```
# example code
# Calculate JCRR using sample dataset J5S1000
JSR(J5S1000)
```

---

LCA

*Latent Class Analysis*

---

### Description

Performs Latent Class Analysis (LCA) on binary response data using the Expectation-Maximization (EM) algorithm. LCA identifies unobserved (latent) subgroups of examinees with similar response patterns, and estimates both the class characteristics and individual membership probabilities.

**Usage**

```
LCA(
  U,
  ncls = 2,
  na = NULL,
  Z = NULL,
  w = NULL,
  maxiter = 100,
  verbose = TRUE,
  beta1 = 1,
  beta2 = 1,
  conf = NULL
)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
ncls	Number of latent classes to identify (between 2 and 20). Default is 2.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.
maxiter	Maximum number of EM algorithm iterations. Default is 100.
verbose	Logical; if TRUE, displays progress during estimation. Default is TRUE.
beta1	Beta distribution parameter 1 for prior density of class reference matrix. Default is 1.
beta2	Beta distribution parameter 2 for prior density of class reference matrix. Default is 1.
conf	Confirmatory IRP matrix (items x ncls) for test equating. Same format as the IRP output. Non-NA values are fixed throughout estimation, NA values are freely estimated. Fixed values must be in the open interval (0, 1). When row names are present, items are matched by label; otherwise by position. Default is NULL (fully exploratory).

**Details**

Latent Class Analysis is a statistical method for identifying unobserved subgroups within a population based on observed response patterns. It assumes that examinees belong to one of several distinct latent classes, and that the probability of a correct response to each item depends on class membership.

The algorithm proceeds by:

1. Initializing class reference probabilities
2. Computing posterior class membership probabilities for each examinee (E-step)

3. Re-estimating class reference probabilities based on these memberships (M-step)
4. Iterating until convergence or reaching the maximum number of iterations

Unlike Item Response Theory (IRT), LCA treats latent variables as categorical rather than continuous, identifying distinct profiles rather than positions on a continuum.

### Value

An object of class "exametrika" and "LCA" containing:

**msg** A character string indicating the model type.

**testlength** Length of the test (number of items).

**nobs** Sample size (number of rows in the dataset).

**Nclass** Number of latent classes specified.

**N\_Cycle** Number of EM algorithm iterations performed.

**converge** Logical value indicating whether the algorithm converged within maxiter iterations

**TRP** Test Reference Profile vector showing expected scores for each latent class. Calculated as the column sum of the estimated class reference matrix.

**LCD** Latent Class Distribution vector showing the number of examinees assigned to each latent class.

**CMD** Class Membership Distribution vector showing the sum of membership probabilities for each latent class.

**Students** Class Membership Profile matrix showing the posterior probability of each examinee belonging to each latent class. The last column ("Estimate") indicates the most likely class assignment.

**IRP** Item Reference Profile matrix where each row represents an item and each column represents a latent class. Values indicate the probability of a correct response for members of that class.

**ItemFitIndices** Fit indices for each item. See also [ItemFit](#).

**TestFitIndices** Overall fit indices for the test. See also [TestFit](#).

### References

Goodman, L. A. (1974). Exploratory latent structure analysis using both identifiable and unidentifiable models. *Biometrika*, 61(2), 215-231.

Lazarsfeld, P. F., & Henry, N. W. (1968). *Latent structure analysis*. Boston: Houghton Mifflin.

### Examples

```
# Fit a Latent Class Analysis model with 5 classes to the sample dataset
result.LCA <- LCA(J15S500, ncls = 5)

# Display the first few rows of student class membership probabilities
head(result.LCA$Students)

# Plot Item Response Profiles (IRP) for items 1-6 in a 2x3 grid
# Shows probability of correct response for each item across classes
```

```

plot(result.LCA, type = "IRP", items = 1:6, nc = 2, nr = 3)

# Plot Class Membership Probabilities (CMP) for students 1-9 in a 3x3 grid
# Shows probability distribution of class membership for each student
plot(result.LCA, type = "CMP", students = 1:9, nc = 3, nr = 3)

# Plot Test Response Profile (TRP) showing expected scores for each class
plot(result.LCA, type = "TRP")

# Plot Latent Class Distribution (LCD) showing class sizes
plot(result.LCA, type = "LCD")

# Compare models with different numbers of classes
# (In practice, you might try more class counts)
lca2 <- LCA(J15S500, ncls = 2)
lca3 <- LCA(J15S500, ncls = 3)
lca4 <- LCA(J15S500, ncls = 4)
lca5 <- LCA(J15S500, ncls = 5)

# Compare BIC values to select optimal number of classes
# (Lower BIC indicates better fit)
data.frame(
  Classes = 2:5,
  BIC = c(
    lca2$TestFitIndices$BIC,
    lca3$TestFitIndices$BIC,
    lca4$TestFitIndices$BIC,
    lca5$TestFitIndices$BIC
  )
)

```

**Description**

Latent dependence Biclustering, which incorporates biclustering and a Bayesian network model.

**Usage**

```

LDB(
  U,
  Z = NULL,
  w = NULL,
  na = NULL,
  ncls = 2,
  method = "R",
  conf = NULL,

```

```

    g_list = NULL,
    adj_list = NULL,
    adj_file = NULL,
    verbose = FALSE,
    beta1 = 1,
    beta2 = 1
)

```

### Arguments

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
Z	Z is a missing indicator matrix of the type matrix or data.frame
w	w is item weight vector
na	na argument specifies the numbers or characters to be treated as missing values.
ncls	number of latent class(rank). The default is 2.
method	specify the model to analyze the data.Local dependence latent class model is set to "C", latent rank model is set "R". The default is "R".
conf	For the confirmatory parameter, you can input either a vector with items and corresponding fields in sequence, or a field membership profile matrix. In the case of the former, the field membership profile matrix will be generated internally. When providing a membership profile matrix, it needs to be either matrix or data.frame. The number of fields(nfld) will be overwrite to the number of columns of this matrix.
g_list	A list compiling graph-type objects for each rank/class.
adj_list	A list compiling matrix-type adjacency matrices for each rank/class.
adj_file	A file detailing the relationships of the graph for each rank/class, listed in the order of starting point, ending point, and rank(class).
verbose	verbose output Flag. default is TRUE
beta1	Beta distribution parameter 1 for prior density. Default is 1.
beta2	Beta distribution parameter 2 for prior density. Default is 1.

### Value

**nobs** Sample size. The number of rows in the dataset.

**testlength** Length of the test. The number of items included in the test.

**msg** A character string indicating the model type.

**Nclass** Optimal number of classes.

**Nfield** Optimal number of fields.

**crr** Correct Response Rate

**ItemLabel** Label of Items

**FieldLabel** Label of Fields

**adj\_list** List of Adjacency matrix used in the model

- g\_list** List of graph object used in the model
- IRP** List of Estimated Parameters. This object is three-dimensional PIRP array, where each dimension represents the number of rank, number of field, and Dmax. Dmax denotes the maximum number of correct response patterns for each field.
- LFD** Latent Field Distribution. see also [plot.exametrika](#)
- LRD** Latent Rank Distribution. see also [plot.exametrika](#)
- FRP** Marginal Field Reference Matrix
- FRPIndex** Index of FRP includes the item location parameters B and Beta, the slope parameters A and Alpha, and the monotonicity indices C and Gamma.
- CCRR\_table** This table is a rearrangement of IRP into a data.frame format for output, consisting of combinations of rank ,field and PIRP.
- TRP** Test Reference Profile
- RMD** Rank Membership Distribution.
- FieldEstimated** Given vector which correspondence between items and the fields.
- ClassEstimated** An index indicating which class a student belongs to, estimated by confirmatory Rankclustering.
- Students** Rank Membership Profile matrix. The s-th row vector of  $\hat{M}_R, \hat{m}_R$ , is the rank membership profile of Student s, namely the posterior probability distribution representing the student's belonging to the respective latent classes. It also includes the rank with the maximum estimated membership probability, as well as the rank-up odds and rank-down odds.
- TestFitIndices** Overall fit index for the test. See also [TestFit](#)

## Examples

```
# Example: Latent Dirichlet Bayesian Network model
# Create field configuration vector based on field assignments
conf <- c(
  1, 6, 6, 8, 9, 9, 4, 7, 7, 7, 5, 8, 9, 10, 10, 9, 9,
  10, 10, 10, 2, 2, 3, 3, 5, 5, 6, 9, 9, 10, 1, 1, 7, 9, 10
)

# Create edge data for the network structure between fields
edges_data <- data.frame(
  "From Field (Parent) >>>" = c(
    6, 4, 5, 1, 1, 4, # Class/Rank 2
    3, 4, 6, 2, 4, 4, # Class/Rank 3
    3, 6, 4, 1, # Class/Rank 4
    7, 9, 6, 7 # Class/Rank 5
  ),
  ">>> To Field (Child)" = c(
    8, 7, 8, 7, 2, 5, # Class/Rank 2
    5, 8, 8, 4, 6, 7, # Class/Rank 3
    5, 8, 5, 8, # Class/Rank 4
    10, 10, 8, 9 # Class/Rank 5
  ),
  "At Class/Rank (Locus)" = c(
    2, 2, 2, 2, 2, 2, # Class/Rank 2
```

```

    3, 3, 3, 3, 3, 3, # Class/Rank 3
    4, 4, 4, 4, # Class/Rank 4
    5, 5, 5, 5 # Class/Rank 5
  )
)

# Save edge data to temporary CSV file
tmp_file <- tempfile(fileext = ".csv")
write.csv(edges_data, file = tmp_file, row.names = FALSE)

# Fit Latent Dirichlet Bayesian Network model
result.LDB <- LDB(
  U = J35S515,
  ncls = 5, # Number of latent classes
  conf = conf, # Field configuration vector
  adj_file = tmp_file # Path to the CSV file
)

# Clean up temporary file
unlink(tmp_file)

# Display model results
print(result.LDB)

# Visualize different aspects of the model
plot(result.LDB, type = "Array") # Show bicluster structure
plot(result.LDB, type = "TRP") # Test Response Profile
plot(result.LDB, type = "LRD") # Latent Rank Distribution
plot(result.LDB,
  type = "RMP", # Rank Membership Profiles
  students = 1:9, nc = 3, nr = 3
)
plot(result.LDB,
  type = "FRP", # Field Reference Profiles
  nc = 3, nr = 2
)
# Field PIRP Profile showing correct answer counts for each rank and field
plot(result.LDB, type = "FieldPIRP")

```

**Description**

performs local dependence latent lank analysis(LD\_LRA) by Shojima(2011)

**Usage**

```
LDLRA(
```

```

    U,
    Z = NULL,
    w = NULL,
    na = NULL,
    ncls = 2,
    method = "R",
    g_list = NULL,
    adj_list = NULL,
    adj_file = NULL,
    verbose = FALSE,
    beta1 = 2,
    beta2 = 2
)

```

### Arguments

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
Z	Z is a missing indicator matrix of the type matrix or data.frame
w	w is item weight vector
na	na argument specifies the numbers or characters to be treated as missing values.
ncls	number of latent class(rank). The default is 2.
method	specify the model to analyze the data. Local dependence latent class model is set to "C", latent rank model is set "R". The default is "R".
g_list	A list compiling graph-type objects for each rank/class.
adj_list	A list compiling matrix-type adjacency matrices for each rank/class.
adj_file	A file detailing the relationships of the graph for each rank/class, listed in the order of starting point, ending point, and rank(class).
verbose	verbose output Flag. default is TRUE
beta1	Beta distribution parameter 1 for prior density of rank reference matrix. Default is 2.
beta2	Beta distribution parameter 2 for prior density of rank reference matrix. Default is 2.

### Details

This function is intended to perform LD-LRA. LD-LRA is an analysis that combines LRA and BNM, and it is used to analyze the network structure among items in the latent rank. In this function, structural learning is not performed, so you need to provide item graphs for each rank as separate files. The file format for this is plain text CSV that includes edges (From, To) and rank numbers.

### Value

**nobs** Sample size. The number of rows in the dataset.  
**msg** A character string indicating the model type.

**testlength** Length of the test. The number of items included in the test.

**crr** correct response ratio

**adj\_list** adjacency matrix list

**g\_list** graph list

**referenceMatrix** Learned Parameters. A three-dimensional array of patterns where item x rank x pattern.

**IRP** Marginal Item Reference Matrix

**IRPIndex** IRP Indices which include Alpha, Beta, Gamma.

**TRP** Test Reference Profile matrix.

**LRD** latent Rank/Class Distribution

**RMD** Rank/Class Membership Distribution

**TestFitIndices** Overall fit index for the test. See also [TestFit](#)

**Estimation\_table** Estimated parameters tables.

**CCRR\_table** Correct Response Rate tables

**Students** Student information. It includes estimated class membership, probability of class membership, RUO, and RDO.

### Examples

```
# Create sample DAG structure with different rank levels
# Format: From, To, Rank
DAG_dat <- matrix(c(
  "From", "To", "Rank",
  "Item01", "Item02", "1", # Simple structure for Rank 1
  "Item01", "Item02", "2", # More complex structure for Rank 2
  "Item02", "Item03", "2",
  "Item01", "Item02", "3", # Additional connections for Rank 3
  "Item02", "Item03", "3",
  "Item03", "Item04", "3"
), ncol = 3, byrow = TRUE)

# Method 1: Directly use graph and adjacency lists
g_list <- list()
adj_list <- list()

for (i in 1:3) {
  adj_R <- DAG_dat[DAG_dat[, 3] == as.character(i), 1:2, drop = FALSE]
  g_tmp <- igraph::graph_from_data_frame(
    d = data.frame(
      From = adj_R[, 1],
      To = adj_R[, 2]
    ),
    directed = TRUE
  )
  adj_tmp <- igraph::as_adjacency_matrix(g_tmp)
  g_list[[i]] <- g_tmp
  adj_list[[i]] <- adj_tmp
}
```

```

}

# Fit Local Dependence Latent Rank Analysis
result.LDLRA1 <- LDLRA(J12S5000,
  ncls = 3,
  g_list = g_list,
  adj_list = adj_list
)

# Plot Item Reference Profiles (IRP) in a 4x3 grid
# Shows the probability patterns of correct responses for each item across ranks
plot(result.LDLRA1, type = "IRP", nc = 4, nr = 3)

# Plot Test Reference Profile (TRP)
# Displays the overall pattern of correct response probabilities across ranks
plot(result.LDLRA1, type = "TRP")

# Plot Latent Rank Distribution (LRD)
# Shows the distribution of students across different ranks
plot(result.LDLRA1, type = "LRD")

```

---

LDLRA\_PBIL

*Structure Learning for LDLRA by PBIL algorithm*


---

## Description

Generating DAG list from data using Population-Based Incremental learning

## Usage

```

LDLRA_PBIL(
  U,
  Z = NULL,
  w = NULL,
  na = NULL,
  seed = 123,
  ncls = 2,
  method = "R",
  population = 20,
  Rs = 0.5,
  Rm = 0.002,
  maxParents = 2,
  maxGeneration = 100,
  successivelimit = 5,
  elitism = 0,
  alpha = 0.05,
  estimate = 1,

```

```

    filename = NULL,
    verbose = TRUE,
    beta1 = 2,
    beta2 = 2
)

```

### Arguments

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
Z	Z is a missing indicator matrix of the type matrix or data.frame
w	w is item weight vector
na	na argument specifies the numbers or characters to be treated as missing values.
seed	seed for random.
ncIs	number of latent class(rank). The default is 2.
method	specify the model to analyze the data. Local dependence latent class model is set to "C", latent rank model is set "R". The default is "R".
population	Population size. The default is 20
Rs	Survival Rate. The default is 0.5
Rm	Mutation Rate. The default is 0.002
maxParents	Maximum number of edges emanating from a single node. The default is 2.
maxGeneration	Maximum number of generations.
successiveLimit	Termination conditions. If the optimal individual does not change for this number of generations, it is considered to have converged.
elitism	Number of elites that remain without crossover when transitioning to the next generation.
alpha	Learning rate. The default is 0.05
estimate	In PBIL for estimating the adjacency matrix, specify by number from the following four methods: 1. Optimal adjacency matrix, 2. Rounded average of individuals in the last generation, 3. Rounded average of survivors in the last generation, 4. Rounded generational gene of the last generation. The default is 1.
filename	Specify the filename when saving the generated adjacency matrix in CSV format. The default is null, and no output is written to the file.
verbose	verbose output Flag. default is TRUE
beta1	Beta distribution parameter 1 for prior density. Default is 2.
beta2	Beta distribution parameter 2 for prior density. Default is 2.

### Details

This function performs structural learning for each classes by using the Population-Based Incremental Learning model(PBIL) proposed by Fukuda et al.(2014) within the genetic algorithm framework. Instead of learning the adjacency matrix itself, the 'genes of genes' that generate the adjacency matrix are updated with each generation. For more details, please refer to Fukuda(2014) and Section 9.4.3 of the text(Shojima,2022).

**Value**

- nobs** Sample size. The number of rows in the dataset.
- testlength** Length of the test. The number of items included in the test.
- crr** correct response ratio
- adj\_list** adjacency matrix list
- g\_list** graph list
- referenceMatrix** Learned Parameters. A three-dimensional array of patterns where item x rank x pattern.
- IRP** Marginal Item Reference Matrix
- IRPIndex** IRP Indices which include Alpha, Beta, Gamma.
- TRP** Test Reference Profile matrix.
- LRD** latent Rank/Class Distribution
- RMD** Rank/Class Membership Distribution
- TestFitIndices** Overall fit index for the test. See also [TestFit](#)
- Estimation\_table** Estimated parameters tables.
- CCRR\_table** Correct Response Rate tables
- Students** Student information. It includes estimated class membership, probability of class membership, RUO, and RDO.

**References**

Fukuda, S., Yamanaka, Y., & Yoshihiro, T. (2014). A Probability-based evolutionary algorithm with mutations to learn Bayesian networks. *International Journal of Artificial Intelligence and Interactive Multimedia*, 3, 7–13. DOI: 10.9781/ijimai.2014.311

**Examples**

```
# Perform Structure Learning for LDLRA using PBIL algorithm
# This process may take considerable time due to evolutionary optimization
result.LDLRA.PBIL <- LDLRA_PBIL(J35S515,
  seed = 123, # Set random seed for reproducibility
  ncls = 5, # Number of latent ranks
  maxGeneration = 10,
  method = "R", # Use rank model (vs. class model)
  elitism = 1, # Keep best solution in each generation
  successiveLimit = 15 # Convergence criterion
)

# Examine the learned network structure
# Plot Item Response Profiles showing item patterns across ranks
plot(result.LDLRA.PBIL, type = "IRP", nc = 4, nr = 3)

# Plot Test Response Profile showing overall response patterns
plot(result.LDLRA.PBIL, type = "TRP")

# Plot Latent Rank Distribution showing student distribution
```

```
plot(result.LDLRA.PBIL, type = "LRD")
```

---

LD\_param\_est

*LDparam set*

---

### Description

A function that extracts only the estimation of graph parameters after the rank estimation is completed.

### Usage

```
LD_param_est(
  tmp,
  adj_list,
  classRefMat,
  ncls,
  smoothpost,
  beta1 = 2,
  beta2 = 2
)
```

### Arguments

tmp	tmp
adj_list	adj_list
classRefMat	values returned from emclus
ncls	ncls
smoothpost	smoothpost
beta1	Beta distribution parameter 1 for prior density. Default is 2.
beta2	Beta distribution parameter 2 for prior density. Default is 2.

---

LogisticModel

*Four-Parameter Logistic Model*

---

### Description

The four-parameter logistic model is a model where one additional parameter  $d$ , called the upper asymptote parameter, is added to the 3PLM.

### Usage

```
LogisticModel(a = 1, b, c = 0, d = 1, theta)
```

**Arguments**

a	slope parameter
b	location parameter
c	lower asymptote parameter
d	upper asymptote parameter
theta	ability parameter

**Value**

Returns a numeric vector of probabilities between c and d, representing the probability of a correct response given the ability level theta. The probability is calculated using the formula:  $P(\theta) = c + \frac{(d-c)}{1+e^{-a(\theta-b)}}$

---

longdataFormat	<i>Long Format Data Conversion</i>
----------------	------------------------------------

---

**Description**

A function to reshape long data into a dataset suitable for exametrika.

**Usage**

```
longdataFormat(
  data,
  na = NULL,
  Sid = NULL,
  Qid = NULL,
  Resp = NULL,
  w = NULL,
  response.type = NULL,
  CA = NULL
)
```

**Arguments**

data	is a data matrix of the type matrix or data.frame. This must contain at least three columns to identify the student, the item, and the response. Additionally, it can include a column for the weight of the items.
na	na argument specifies the numbers or characters to be treated as missing values.
Sid	Specify the column number containing the student ID label vector.
Qid	Specify the column number containing the Question label vector.
Resp	Specify the column number containing the Response value vector.
w	Specify the column number containing the weight vector.

<code>response.type</code>	Character string specifying the type of response data: "binary" for dichotomous data, "ordinal" for ordered polytomous data, "rated" for polytomous data with correct answers, "nominal" for unordered polytomous data. If NULL (default), the type is automatically detected.
<code>CA</code>	A numeric vector specifying the correct answers for rated polytomous data. Required when <code>response.type</code> is "rated".

### Value

- U** For binary response data. A matrix with rows representing the sample size and columns representing the number of items, where elements are either 0 or 1.  $u_{ij} = 1$  indicates that student  $i$  correctly answered item  $j$ , while  $u_{ij} = 0$  means that student  $i$  answered item  $j$  incorrectly.
- Q** For polytomous response data. A matrix with rows representing the sample size and columns representing the number of items, where elements are non-negative integers. When input data is in factor format, the factor levels are converted to consecutive integers starting from 1.
- ID** The ID label given by the designated column or function.
- ItemLabel** The item names given by the provided column names or function.
- Z** Missing indicator matrix.  $z_{ij} = 1$  indicates that item  $j$  is presented to Student  $i$ , while  $z_{ij} = 0$  indicates item  $j$  is NOT presented to Student  $i$ .
- w** Item weight vector
- response.type** Character string indicating the type of response data: "binary", "ordinal", "rated", or "nominal"
- CategoryLabel** List containing the original factor labels when polytomous responses are provided as factors. NULL if no factor data is present.
- categories** Numeric vector containing the number of response categories for each item.
- CA** For rated polytomous data, a numeric vector of correct answers. NULL for other types.

### Description

A general function for estimating Latent Rank Analysis across different response types. This function automatically dispatches to the appropriate method based on the response type:

- For binary data (`LRA.binary`): Analysis using either SOM or GTM method
- For ordinal data (`LRA.ordinal`): Analysis using the GTM method with category thresholds
- For rated data (`LRA.rated`): Analysis using the GTM method with rating categories

Latent Rank Analysis identifies underlying rank structures in test data and assigns examinees to these ranks based on their response patterns.

**Usage**

```
LRA(U, ...)  
  
## Default S3 method:  
LRA(U, na = NULL, Z = NULL, w = NULL, ...)  
  
## S3 method for class 'binary'  
LRA(  
  U,  
  nrank = 2,  
  method = "GTM",  
  mic = FALSE,  
  maxiter = 100,  
  BIC.check = FALSE,  
  seed = NULL,  
  verbose = FALSE,  
  beta1 = 1,  
  beta2 = 1,  
  conf = NULL,  
  ...  
)  
  
## S3 method for class 'ordinal'  
LRA(  
  U,  
  nrank = 2,  
  mic = FALSE,  
  maxiter = 100,  
  trapezoidal = 0,  
  eps = 1e-04,  
  verbose = FALSE,  
  ...  
)  
  
## S3 method for class 'rated'  
LRA(  
  U,  
  nrank = 2,  
  mic = FALSE,  
  maxiter = 100,  
  trapezoidal = 0,  
  eps = 1e-04,  
  minFreqRatio = 0,  
  verbose = FALSE,  
  ...  
)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
...	Additional arguments passed to specific methods.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. 1 indicates observed values, 0 indicates missing values.
w	Item weight vector.
nrank	Number of latent ranks to estimate. Must be between 2 and 20.
method	For binary data only. Either "SOM" (Self-Organizing Maps) or "GTM" (Gaussian Topographic Mapping). Default is "GTM".
mic	Logical; if TRUE, forces Item Reference Profiles to be monotonically increasing. Default is FALSE.
maxiter	Maximum number of iterations for estimation. Default is 100.
BIC.check	For binary data with SOM method only. If TRUE, convergence is checked using BIC values. Default is FALSE.
seed	For binary data with SOM method only. Random seed for reproducibility.
verbose	Logical; if TRUE, displays detailed progress during estimation. Default is TRUE.
beta1	Beta distribution parameter 1 for prior density of rank reference matrix (GTM method only). Default is 1.
beta2	Beta distribution parameter 2 for prior density of rank reference matrix (GTM method only). Default is 1.
conf	Confirmatory IRP matrix (items x nrank) for test equating. Same format as the IRP output. Non-NA values are fixed throughout estimation, NA values are freely estimated. Fixed values must be in the open interval (0, 1). When row names are present, items are matched by label; otherwise by position. Default is NULL (fully exploratory).
trapezoidal	Specifies the height of both tails when using a trapezoidal prior distribution. Must be less than 1/nrank. The default value is 0, which results in a uniform prior distribution.
eps	Convergence threshold for parameter updates. Default is 1e-4.
minFreqRatio	Minimum frequency ratio for response categories (default = 0). Categories with occurrence rates below this threshold will be excluded from analysis. For example, if set to 0.1, response categories that appear in less than 10% of responses for an item will be omitted.

**Value**

A list of class "exametrika" and the specific subclass (e.g., "LRA", "LRAordinal", "LRArated") containing the following common elements:

**msg** A character string indicating the model type.

**testlength** Length of the test (number of items).

- nobs** Sample size (number of rows in the dataset).
- Nrank** Number of latent ranks specified.
- N\_Cycle** Number of EM algorithm iterations performed.
- converge** Logical value indicating whether the algorithm converged within maxiter iterations
- TRP** Test Reference Profile vector showing expected scores at each rank.
- LRD** Latent Rank Distribution vector showing the number of examinees at each rank.
- RMD** Rank Membership Distribution vector showing the sum of probabilities for each rank.
- Students** Rank Membership Profile matrix showing the posterior probabilities of examinees belonging to each rank, along with their estimated ranks and odds ratios.
- ItemFitIndices** Fit indices for each item. See also [ItemFit](#).
- TestFitIndices** Overall fit indices for the test. See also [TestFit](#).

Each subclass returns additional specific elements, detailed in their respective documentation.

For binary data (`LRA.binary`), the returned list additionally includes:

- IRP** Item Reference Profile matrix showing the probability of correct response for each item across different ranks.
- IRPIndex** Item Response Profile indices including the location parameters B and Beta, slope parameters A and Alpha, and monotonicity indices C and Gamma.

For ordinal data (`LRA.ordinal`), the returned list additionally includes:

- msg** A character string indicating the model type.
- converge** Logical value indicating whether the algorithm converged within maxiter iterations
- ScoreReport** Descriptive statistics of test performance, including sample size, test length, central tendency, variability, distribution characteristics, and reliability.
- ItemReport** Basic statistics for each item including category proportions and item-total correlations.
- ICBR** Item Category Boundary Reference matrix showing cumulative probabilities for rank-category combinations.
- ICRP** Item Category Reference Profile matrix showing probability of response in each category by rank.
- ScoreRankCorr** Spearman's correlation between test scores and estimated ranks.
- RankQuantCorr** Spearman's correlation between estimated ranks and quantile groups.
- ScoreRank** Contingency table of raw scores by estimated ranks.
- ScoreMembership** Expected rank memberships for each raw score.
- RankQuantile** Cross-tabulation of rank frequencies and quantile groups.
- MembQuantile** Cross-tabulation of rank membership probabilities and quantile groups.
- CatQuant** Response patterns across item categories and quantile groups.

For rated data (`LRA.rated`), the returned list additionally includes:

- msg** A character string indicating the model type.

- converge** Logical value indicating whether the algorithm converged within maxiter iterations
- ScoreReport** Descriptive statistics of test performance, including sample size, test length, central tendency, variability, distribution characteristics, and reliability.
- ItemReport** Basic statistics for each item including category proportions and item-total correlations.
- ICRP** Item Category Reference Profile matrix showing probability of response in each category by rank.
- ScoreRankCorr** Spearman's correlation between test scores and estimated ranks.
- RankQuantCorr** Spearman's correlation between estimated ranks and quantile groups.
- ScoreRank** Contingency table of raw scores by estimated ranks.
- ScoreMembership** Expected rank memberships for each raw score.
- RankQuantile** Cross-tabulation of rank frequencies and quantile groups.
- MembQuantile** Cross-tabulation of rank membership probabilities and quantile groups.
- ItemQuantileRef** Reference values for each item across quantile groups.
- CatQuant** Response patterns across item categories and quantile groups.

### Binary Data Method

LRA.binary analyzes dichotomous (0/1) response data using either Self-Organizing Maps (SOM) or Gaussian Topographic Mapping (GTM).

### Ordinal Data Method

LRA.ordinal analyzes ordered categorical data with multiple thresholds, such as Likert-scale responses or graded items.

### Rated Data Method

LRA.rated analyzes data with ratings assigned to each response, such as partially-credited items or preference scales where response categories have different weights.

### See Also

[plot.exametrika](#) for visualizing LRA results.

### Examples

```
# Binary data example
# Fit a Latent Rank Analysis model with 6 ranks to binary data
result.LRA <- LRA(J15S500, nrank = 6)

# Display the first few rows of student rank membership profiles
head(result.LRA$Students)

# Plot Item Reference Profiles (IRP) for the first 6 items
plot(result.LRA, type = "IRP", items = 1:6, nc = 2, nr = 3)
```

```

# Plot Test Reference Profile (TRP) showing expected scores at each rank
plot(result.LRA, type = "TRP")

# Ordinal data example
# Fit a Latent Rank Analysis model with 3 ranks to ordinal data
result.LRAord <- LRA(J15S3810, nrank = 3, mic = TRUE)

# Plot score distributions
plot(result.LRAord, type = "ScoreFreq")
plot(result.LRAord, type = "ScoreRank")

# Plot category response patterns for items 1-6
plot(result.LRAord, type = "ICBR", items = 1:6, nc = 3, nr = 2)
plot(result.LRAord, type = "ICRP", items = 1:6, nc = 3, nr = 2)

# Rated data example
# Fit a Latent Rank Analysis model with 10 ranks to rated data
result.LRARated <- LRA(J35S5000, nrank = 10, mic = TRUE)

# Plot score distributions
plot(result.LRARated, type = "ScoreFreq")
plot(result.LRARated, type = "ScoreRank")

# Plot category response patterns for items 1-6
plot(result.LRARated, type = "ICRP", items = 1:6, nc = 3, nr = 2)

```

---

maxParents\_penalty      *Utility function for searching DAG*

---

### Description

Function to limit the number of parent nodes

### Usage

```
maxParents_penalty(vec, testlength, maxParents)
```

### Arguments

vec	gene Vector corresponding to the upper triangular of the adjacency matrix
testlength	test length. In this context it means a number of nodes.
maxParents	Upper limit of number of nodes.

## Details

When generating an adjacency matrix using GA, the number of edges coming from a single node should be limited to 2 or 3. This is because if there are too many edges, it becomes difficult to interpret in practical applications. This function works to adjust the sampling of the randomly generated adjacency matrix so that the column sum of the upper triangular elements fits within the set limit.

---

MutualInformation      *Mutual Information*

---

## Description

Mutual Information is a measure that represents the degree of interdependence between two items. This function is applicable to both binary and polytomous response data. The measure is calculated using the joint probability distribution of responses between item pairs and their marginal probabilities.

## Usage

```
MutualInformation(U, na = NULL, Z = NULL, w = NULL, base = 2)

## Default S3 method:
MutualInformation(U, na = NULL, Z = NULL, w = NULL, base = 2)

## S3 method for class 'binary'
MutualInformation(U, na = NULL, Z = NULL, w = NULL, base = 2)

## S3 method for class 'ordinal'
MutualInformation(U, na = NULL, Z = NULL, w = NULL, base = 2)
```

## Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.
base	The base for the logarithm. Default is 2. For polytomous data, you can use "V" to set the base to min(rows, columns), "e" for natural logarithm (base e), or any other number to use that specific base.

## Details

For binary data, the following formula is used:

$$MI_{jk} = p_{00} \log_2 \frac{p_{00}}{(1-p_j)(1-p_k)} + p_{01} \log_2 \frac{p_{01}}{(1-p_j)p_k} + p_{10} \log_2 \frac{p_{10}}{p_j(1-p_k)} + p_{11} \log_2 \frac{p_{11}}{p_j p_k}$$

Where:

- $p_{00}$  is the joint probability of incorrect responses to both items j and k
- $p_{01}$  is the joint probability of incorrect response to item j and correct to item k
- $p_{10}$  is the joint probability of correct response to item j and incorrect to item k
- $p_{11}$  is the joint probability of correct responses to both items j and k

For polytomous data, the following formula is used:

$$MI_{jk} = \sum_{j=1}^{C_j} \sum_{k=1}^{C_k} p_{jk} \log \frac{p_{jk}}{p_j \cdot p_k}$$

The base of the logarithm can be the number of rows, number of columns, min(rows, columns), base-10 logarithm, natural logarithm (e), etc.

## Value

A matrix of mutual information values with exametrika class. Each element (i,j) represents the mutual information between items i and j, measured in bits. Higher values indicate stronger inter-dependence between items.

## Note

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

## Examples

```
# example code
# Calculate Mutual Information using sample dataset J15S500
MutualInformation(J15S500)
```

## Description

The Number-Right Score (NRS) function calculates the weighted sum of correct responses for each examinee. This function is applicable only to binary response data.

For each examinee, the score is computed as:

$$NRS_i = \sum_{j=1}^J z_{ij} u_{ij} w_j$$

where:

- $z_{ij}$  is the missing response indicator (0/1)
- $u_{ij}$  is the response (0/1)
- $w_j$  is the item weight

## Usage

```
nrs(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
nrs(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
nrs(U, na = NULL, Z = NULL, w = NULL)
```

## Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

## Value

A numeric vector containing the Number-Right Score for each examinee. The score represents the weighted sum of correct answers, where:

- Maximum score is the sum of all item weights
- Minimum score is 0
- Missing responses do not contribute to the score

## Note

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

**Examples**

```
# using sample dataset
nrs(J15S500)
```

---

```
objective_function_IRT
```

*Log-likelihood function used in the Maximization Step (M-Step).*

---

**Description**

Log-likelihood function used in the Maximization Step (M-Step).

**Usage**

```
objective_function_IRT(lambda, model, qjtrue, qjfalse, quadrature)
```

**Arguments**

lambda	item parameter vector
model	2,3,or 4 PL
qjtrue	correct resp pattern
qjfalse	incorrect resp pattern
quadrature	Pattern of a segmented normal distribution.

---

OmegaCoefficient	<i>Omega Coefficient</i>
------------------	--------------------------

---

**Description**

This function computes Tau-Congeneric Measurement, also known as McDonald's tau coefficient, for a given data set.

**Usage**

```
OmegaCoefficient(x, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

x	This should be a data matrix or a Covariance/Phi/Tetrachoric matrix.
na	This parameter identifies the numbers or characters that should be treated as missing values when 'x' is a data matrix.
Z	This parameter represents a missing indicator matrix. It is only needed if 'x' is a data matrix.
w	This parameter is an item weight vector. It is only required if 'x' is a data matrix.

**Value**

For a correlation/covariance matrix input, returns a single numeric value representing the omega coefficient. For a data matrix input, returns a list with three components:

**OmegaCov** Omega coefficient calculated from covariance matrix

**OmegaPhi** Omega coefficient calculated from phi coefficient matrix

**OmegaTetrachoric** Omega coefficient calculated from tetrachoric correlation matrix

**References**

McDonald, R. P. (1999). Test theory: A unified treatment. Erlbaum.

---

passage

*Passage Rate of Student*

---

**Description**

The Passage Rate for each student is calculated as their Number-Right Score (NRS) divided by the number of items presented to them. This function is applicable only to binary response data.

The passage rate is calculated as:

$$P_i = \frac{\sum_{j=1}^J z_{ij} u_{ij} w_j}{\sum_{j=1}^J z_{ij}}$$

where:

- $z_{ij}$  is the missing response indicator (0/1)
- $u_{ij}$  is the response (0/1)
- $w_j$  is the item weight

**Usage**

```
passage(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
passage(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
passage(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

A numeric vector containing the passage rate for each student. Values range from 0 to 1 (or maximum weight) where:

- 1: Perfect score on all attempted items
- 0: No correct answers
- NA: No items attempted

**Note**

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

The passage rate accounts for missing responses by only considering items that were actually presented to each student. This provides a fair comparison between students who attempted different numbers of items.

**Examples**

```
# using sample dataset
passage(J15S500)
```

---

percentile

*Student Percentile Ranks*

---

**Description**

The percentile function calculates each student's relative standing in the group, expressed as a percentile rank (1-100). This function is applicable only to binary response data.

The percentile rank indicates the percentage of scores in the distribution that fall below a given score. For example, a percentile rank of 75 means the student performed better than 75% of the group.

**Usage**

```
percentile(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
percentile(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
percentile(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

A numeric vector of percentile ranks (1-100) for each student, where:

- 100: Highest performing student(s)
- 50: Median performance
- 1: Lowest performing student(s)

**Note**

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

Percentile ranks are calculated using the empirical cumulative distribution function of standardized scores. Tied scores receive the same percentile rank. The values are rounded up to the nearest integer to provide ranks from 1 to 100.

**Examples**

```
# using sample dataset
percentile(J5S10)
```

---

 PhiCoefficient

*Phi-Coefficient*


---

**Description**

The phi coefficient is the Pearson's product moment correlation coefficient between two binary items. This function is applicable only to binary response data. The coefficient ranges from -1 to 1, where 1 indicates perfect positive correlation, -1 indicates perfect negative correlation, and 0 indicates no correlation.

**Usage**

```
PhiCoefficient(U, na = NULL, Z = NULL, w = NULL)

## Default S3 method:
PhiCoefficient(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'binary'
PhiCoefficient(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

A matrix of phi coefficients with exametrika class. Each element (i,j) represents the phi coefficient between items i and j. The matrix is symmetric with ones on the diagonal.

**Note**

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

**Examples**

```
# example code
# Calculate Phi-Coefficient using sample dataset J15S500
PhiCoefficient(J15S500)
```

---

plot.exametrika      *Plot Method for Objects of Class "exametrika"*

---

**Description**

Creates visualizations for objects with class "exametrika". The calculation results of the exametrika package have an exametrika class attribute, along with the specific analysis model class (IRT, GRM, LCA, LRA, Biclustering, Biclustering\_IRM, LDLRA, LDB, BINET). Each model has its own compatible plot types, accessible by specifying the 'type' parameter.

**Usage**

```
## S3 method for class 'exametrika'
plot(
  x,
  type = c("IRF", "TRF", "IIF", "TIF", "IIC", "ICC", "TIC", "IRP", "TRP", "LCD", "CMP",
    "FRP", "RMP", "LRD", "Array", "CRV", "RRV", "FCRP", "FCBR", "ScoreField",
    "FieldPIRP", "LDPSR", "ScoreFreq", "ScoreRank", "ICRP", "ICBR"),
  items = NULL,
  students = NULL,
  nc = 1,
  nr = 1,
  overlay = FALSE,
  colors = NULL,
  cell_width = 3,
  cell_height = 1,
  filename = NULL,
  width = 800,
  height = 600,
  dpi = 300,
  stat = "mean",
  style = "line",
  ...
)
```

**Arguments**

x	An object of class "exametrika"
type	Character string specifying the plot type. Available types vary by model: <b>IRF, ICC</b> Item Response Function. Also known as 'ICC' (Item Characteristic Curve). <b>TRF</b> Test Response Function. <b>IIF, IIC</b> Item Information Function. Also known as 'IIC' (Item Information Curve). <b>TIF, TIC</b> Test Information Function. Also known as 'TIC' (Test Information Curve). <b>IRP</b> Item Reference Profile. Line graph with items and latent classes/ranks on the horizontal axis, and membership probability on the vertical axis. <b>CRV, RRV</b> Class/Rank Reference Vector. Plots correct answer rates for each class or rank, with fields on the horizontal axis and correct answer rates on the vertical axis. <b>TRP</b> Test Reference Profile. Shows latent classes/ranks on the horizontal axis, displaying members per class/rank as a bar graph and expected test scores as a line graph. <b>LCD</b> Latent Class Distribution. Displays latent classes on the horizontal axis, showing members per class as a bar graph and cumulative membership probability as a line.

	<b>LRD</b> Latent Rank Distribution. Similar to LCD but with ranks instead of classes on the horizontal axis.
	<b>CMP</b> Class Membership Profile. Line graph showing class membership probabilities of students.
	<b>RMP</b> Rank Membership Profile. Similar to CMP but with ranks instead of classes.
	<b>ScoreFreq</b> Frequency polygon of score distribution with rank thresholds.
	<b>ScoreRank</b> Heatmap of score membership probabilities for each rank.
	<b>ICRP</b> Visualizes ranks (x-axis) versus category response probabilities (y-axis).
	<b>ICBR</b> Visualizes ranks (x-axis) versus cumulative category probabilities (y-axis).
	<b>FRP</b> Field Reference Profile. Shows correspondence between fields and latent classes/ranks.
	<b>Array</b> Array plot for Biclustering/Rankclustering. Colored matrix cells where darker cells indicate larger values.
	<b>FieldPIRP</b> Shows correct response rates by number of correct answers in parent fields. Only available for LDB model.
	<b>LDPSR</b> Latent Dependence Passing Student Rate. Compares passing rates of parent and child classes.
	<b>FCRP</b> Field Category Response Profile. Category probability plot per field for polytomous Biclustering. Use <code>style</code> parameter to choose between "line" and "bar" display.
	<b>FCBR</b> Field Cumulative Boundary Reference. Boundary probability plot per field (ordinal Biclustering only).
	<b>ScoreField</b> Heatmap of expected scores across fields and latent classes/ranks for polytomous Biclustering.
<code>items</code>	Numeric vector specifying which items to plot. If NULL, all items are included. When type is "IIF"/"IIC", specifying 0 will produce a TIF/TIC for the entire test.
<code>students</code>	Numeric vector specifying which students to plot. If NULL, all students are included.
<code>nc</code>	Integer specifying the number of columns for multiple plots. Default is 1.
<code>nr</code>	Integer specifying the number of rows for multiple plots. Default is 1.
<code>overlay</code>	Logical. If TRUE, elements such as IRFs will be overlaid on a single plot. Default is FALSE.
<code>colors</code>	Character vector specifying custom color palette. If NULL, default colorblind-friendly palette is used. For array plots, the first color should be for missing values, followed by response category colors.
<code>cell_width</code>	Numeric value specifying the width of each cell in array plots. Default is 3.
<code>cell_height</code>	Numeric value specifying the height of each cell in array plots. Default is 1.
<code>filename</code>	Character string specifying output filename. If NULL, plot is displayed on screen. Supported formats: png, pdf, jpeg, jpg. Format determined by file extension.
<code>width</code>	Numeric value specifying plot width in pixels (for png/jpeg) or inches (for pdf). Default is 800.

height	Numeric value specifying plot height in pixels (for png/jpeg) or inches (for pdf). Default is 600.
dpi	Numeric value specifying resolution in dots per inch for raster formats (png/jpeg). Default is 300.
stat	Character string specifying the summary statistic for polytomous FRP and RRV plots. One of "mean" (default), "median", or "mode".
style	Character string specifying the display style for FCRP plots. One of "line" (default) or "bar" (stacked bar chart).
...	Additional arguments passed to plotting functions.

### Details

Each model class supports specific plot types:

**IRT** Supports "IRF"/"ICC", "TRF", "IIF"/"IIC", "TIF"/"TIC"

**GRM** Supports "IRF"/"ICC", "IIF"/"IIC", "TIF"/"TIC"

**LCA** Supports "IRP", "FRP", "TRP", "LCD", "CMP"

**LRA** Supports "IRP", "FRP", "TRP", "LRD", "RMP"

**LRAordinal** Supports "ScoreFreq", "ScoreRank", "ICRP", "ICBR", "RMP"

**LRArated** Supports "ScoreFreq", "ScoreRank", "ICRP", "RMP"

**Biclustering** Supports "FRP", "TRP", "LCD", "LRD", "CMP", "RMP", "CRV", "RRV", "Array"

**ordinalBiclustering** Supports "FRP", "FCRP", "FCBR", "LCD", "LRD", "CMP", "RMP", "Array", "ScoreField", "RRV"

**nominalBiclustering** Supports "FRP", "FCRP", "LCD", "LRD", "CMP", "Array", "ScoreField", "RRV"

**Biclustering\_IRM** Supports "FRP", "TRP", "Array"

**LDLRA** Supports "IRP", "TRP", "LRD", "RMP"

**LDB** Supports "FRP", "TRP", "LRD", "RMP", "Array", "FieldPIRP"

**BINET** Supports "FRP", "TRP", "LRD", "RMP", "Array", "LDPSR"

### Value

Produces visualizations based on the model class and specified type:

**IRT models** IRF (Item Response Function), TRF (Test Response Function), IIF (Item Information Function), TIF (Test Information Function)

**LCA/LRA models** IRP (Item Reference Profile), TRP (Test Reference Profile), LCD/LRD (Latent Class/Rank Distribution), CMP/RMP (Class/Rank Membership Profile)

**Biclustering/Biclustering\_IRM models** Array plots showing clustering patterns, FRP, TRP, etc.

**LDLRA/LDB/BINET models** Network and profile plots specific to each model

**Examples**

```
## Not run:
# IRT model example
irt_result <- exametrika::IRT(J15S500)
plot(irt_result, type = "IRF", items = 1:5)
plot(irt_result, type = "TIF")

# LCA model example
lca_result <- exametrika::LCA(U)
plot(lca_result, type = "IRP")
plot(lca_result, type = "LCD")

## End(Not run)

# Array plot with custom output
biclustering_result <- exametrika::Biclustering(J35S515)
# Custom colors and file output
my_colors <- c("#404040", "#E69F00", "#56B4E9", "#009E73", "#F0E442")
plot(biclustering_result, type = "Array", colors = my_colors)
```

---

polychoric

*Polychoric Correlation*

---

**Description**

Calculate the polychoric correlation coefficient between two polytomous (categorical ordinal) variables. Polychoric correlation estimates the correlation between two theorized normally distributed continuous latent variables from two observed ordinal variables.

**Usage**

```
polychoric(x, y)
```

**Arguments**

x	A polytomous vector (categorical ordinal variable)
y	A polytomous vector (categorical ordinal variable)

**Details**

This function handles missing values (coded as -1 or NA) using pairwise deletion. The estimation uses maximum likelihood approach with Brent's method for optimization.

**Value**

The polychoric correlation coefficient between x and y

**Examples**

```
# Example with simulated data
set.seed(123)
x <- sample(1:5, 100, replace = TRUE)
y <- sample(1:4, 100, replace = TRUE)
polychoric(x, y)
```

---

PolychoricCorrelationMatrix  
*Polychoric Correlation Matrix*

---

**Description**

Polychoric Correlation Matrix

**Usage**

```
PolychoricCorrelationMatrix(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
PolychoricCorrelationMatrix(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'ordinal'
```

```
PolychoricCorrelationMatrix(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

A matrix of polychoric correlations with exametrika class. Each element (i,j) represents the polychoric correlation between items i and j. The matrix is symmetric with ones on the diagonal.

**Examples**

```
# example code
PolychoricCorrelationMatrix(J5S1000)
```

---

polyserial

*Polyserial Correlation*

---

### Description

Calculates the polyserial correlation coefficient between a continuous variable and an ordinal variable.

### Usage

```
polyserial(x, y)
```

### Arguments

x                    A numeric vector representing the continuous variable.  
y                    A numeric vector representing the ordinal variable (must be integer values).

### Details

This function implements Olsson et al.'s ad hoc method for estimating the polyserial correlation coefficient. The method assumes that the continuous variable follows a normal distribution and that the ordinal variable is derived from an underlying continuous normal variable through thresholds.

### Value

A numeric value representing the estimated polyserial correlation coefficient.

### References

U.Olsson, F.Drasgow, and N.Dorans (1982). The polyserial correlation coefficient. *Psychometrika*, 47,337-347.

### Examples

```
n <- 300  
x <- rnorm(n)  
y <- sample(1:5, size = n, replace = TRUE)  
polyserial(x, y)
```

---

print.exametrika      *Print Method for Exametrika Objects*

---

### Description

S3 method for printing objects of class "exametrika". This function formats and displays appropriate summary information based on the specific subclass of the exametrika object. Different types of analysis results (IRT, LCA, network models, etc.) are presented with customized formatting to highlight the most relevant information.

### Usage

```
## S3 method for class 'exametrika'
print(x, digits = 3, ...)
```

### Arguments

x	An object of class "exametrika" with various possible subclasses
digits	Integer indicating the number of decimal places to display. Default is 3.
...	Additional arguments passed to print methods (not currently used)

### Details

The function identifies the specific subclass of the exametrika object and tailors the output accordingly. For most analysis types, the function displays:

- Basic model description and parameters
- Estimation results (e.g., item parameters, latent class profiles)
- Model fit statistics and diagnostics
- Visual representations where appropriate (e.g., graphs for network models, scree plots for dimensionality analysis)

When printing network-based models (LDLRA, LDB, BINET), this function visualizes the network structure using graphs, which can help in interpreting complex relationships between items or latent variables.

### Value

Prints a formatted summary of the exametrika object to the console, with content varying by object subclass:

**TestStatistics** Basic descriptive statistics of the test

**Dimensionality** Eigenvalue analysis results with scree plot

**ItemStatistics** Item-level statistics and psychometric properties

**QitemStatistics** Item statistics for polytomous items

**exametrikaData** Data structure details including response patterns and weights  
**IIAnalysis** Item-item relationship measures (tetrachoric correlations, etc.)  
**CTT** Classical Test Theory reliability measures  
**IRT/GRM** Item parameters, ability estimates, and fit indices  
**LCA/LRA** Class/Rank profiles, distribution information, and model fit statistics  
**Biclustering/Biclustering\_IRM** Cluster profiles, field distributions, and model diagnostics  
**LDLRA/LDB/BINET** Network visualizations, parameter estimates, and conditional probabilities

### Examples

```
# Print IRT analysis results with 4 decimal places
result <- IRT(J15S500)
print(result, digits = 4)

# Print Latent Class Analysis results
result_lca <- LCA(J15S500, ncls = 3)
print(result_lca)
```

---

PSD_item_params	<i>internal functions for PSD of Item parameters</i>
-----------------	--

---

### Description

internal functions for PSD of Item parameters

### Usage

```
PSD_item_params(model, Lambda, quadrature, marginal_posttheta)
```

### Arguments

model	2,3,or 4PL
Lambda	item parameters Matrix
quadrature	quads
marginal_posttheta	marginal post theta

---

 RaschModel

*Rasch Model*


---

### Description

The one-parameter logistic model is a model with only one parameter  $b$ . This model is a 2PLM model in which  $a$  is constrained to 1. This model is also called the Rasch model.

### Usage

```
RaschModel(b, theta)
```

### Arguments

$b$	slope parameter
$\theta$	ability parameter

### Value

Returns a numeric vector of probabilities between 0 and 1, representing the probability of a correct response given the ability level  $\theta$ . The probability is calculated using the formula:  $P(\theta) = \frac{1}{1+e^{-(\theta-b)}}$

---

 ScoreReport

*Generate Score Report for Non-Binary Test Data*


---

### Description

Calculates comprehensive descriptive statistics for a test, including measures of central tendency, variability, distribution shape, and reliability.

### Usage

```
ScoreReport(U, na = NULL, Z = NULL, w = NULL)
```

### Arguments

$U$	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
$na$	Values to be treated as missing values
$Z$	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
$w$	Item weight vector specifying the relative importance of each item

**Details**

This function is intended for non-binary (ordinal or rated) response data. It calculates descriptive statistics for the overall test performance. If binary data is provided, an error message will be displayed.

**Value**

An object of class "exametrika" and "TestStatistics" containing:

**TestLength** Number of items included in the test

**SampleSize** Number of examinees (rows) in the dataset

**Mean** Average score across all examinees

**Median** Median score

**SD** Standard deviation of test scores

**Variance** Variance of test scores

**Skewness** Skewness of the score distribution (measure of asymmetry)

**Kurtosis** Kurtosis of the score distribution (measure of tail extremity)

**Min** Minimum score obtained

**Max** Maximum score obtained

**Range** Difference between maximum and minimum scores

**Alpha** Cronbach's alpha coefficient, a measure of internal consistency reliability

**Examples**

```
# Generate score report for sample ordinal data
ScoreReport(J15S3810)
```

```
# Example with rated data
ScoreReport(J35S5000)
```

---

slopeprior

*Prior distribution function with respect to the slope.*

---

**Description**

Prior distribution function with respect to the slope.

**Usage**

```
slopeprior(a, m, s, const = 1e-15)
```

**Arguments**

a	slope coefficient
m	prior parameter to be set
s	prior parameter to be set
const	A very small constant

---

softmax	<i>softmax function</i>
---------	-------------------------

---

**Description**

to avoid overflow

**Usage**

```
softmax(x)
```

**Arguments**

x	numeric vector
---	----------------

---

sscore	<i>Standardized Score</i>
--------	---------------------------

---

**Description**

The standardized score (z-score) indicates how far a student's performance deviates from the mean in units of standard deviation. This function is applicable only to binary response data.

The score is calculated by standardizing the passage rates:

$$Z_i = \frac{r_i - \bar{r}}{\sigma_r}$$

where:

- $r_i$  is student i's passage rate
- $\bar{r}$  is the mean passage rate
- $\sigma_r$  is the standard deviation of passage rates

**Usage**

```
sscore(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
sscore(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
sscore(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

A numeric vector of standardized scores for each student. The scores follow a standard normal distribution with:

- Mean = 0
- Standard deviation = 1
- Approximately 68% of scores between -1 and 1
- Approximately 95% of scores between -2 and 2
- Approximately 99% of scores between -3 and 3

**Note**

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

The standardization allows for comparing student performance across different tests or groups. A positive score indicates above-average performance, while a negative score indicates below-average performance.

**Examples**

```
# using sample dataset
sscore(J5S10)
```

---

 stanine

*Stanine Scores*


---

**Description**

The Stanine (Standard Nine) scoring system divides students into nine groups based on a normalized distribution. This function is applicable only to binary response data.

These groups correspond to the following percentile ranges:

- Stanine 1: lowest 4% (percentiles 1-4)
- Stanine 2: next 7% (percentiles 5-11)
- Stanine 3: next 12% (percentiles 12-23)

- Stanine 4: next 17% (percentiles 24-40)
- Stanine 5: middle 20% (percentiles 41-60)
- Stanine 6: next 17% (percentiles 61-77)
- Stanine 7: next 12% (percentiles 78-89)
- Stanine 8: next 7% (percentiles 90-96)
- Stanine 9: highest 4% (percentiles 97-100)

### Usage

```
stanine(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
stanine(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
stanine(U, na = NULL, Z = NULL, w = NULL)
```

### Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

### Value

A list containing two elements:

**stanine** The score boundaries for each stanine level

**stanineScore** The stanine score (1-9) for each student

### Note

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

Stanine scores provide a normalized scale with:

- Mean = 5
- Standard deviation = 2
- Scores range from 1 to 9
- Score of 5 represents average performance

## References

Angoff, W. H. (1984). Scales, norms, and equivalent scores. Educational Testing Service. (Reprint of chapter in R. L. Thorndike (Ed.) (1971) Educational Measurement (2nd Ed.). American Council on Education.

## Examples

```
result <- stanine(J15S500)
# View score boundaries
result$stanine
# View individual scores
result$stanineScore
```

---

StrLearningGA\_BNM      *StrLearningGA\_BNM (Deprecated)*

---

## Description

This function has been renamed to [BNM\\_GA](#). Please use [BNM\\_GA](#) instead.

## Usage

```
StrLearningGA_BNM(...)
```

## Arguments

...                    All arguments passed to [BNM\\_GA](#)

---

StrLearningPBIL\_BNM      *StrLearningPBIL\_BNM (Deprecated)*

---

## Description

This function has been renamed to [BNM\\_PBIL](#). Please use [BNM\\_PBIL](#) instead.

## Usage

```
StrLearningPBIL_BNM(...)
```

## Arguments

...                    All arguments passed to [BNM\\_PBIL](#)

---

StrLearningPBIL\_LDLRA *StrLearningPBIL\_LDLRA (Deprecated)*

---

### Description

This function has been renamed to [LDLRA\\_PBIL](#). Please use LDLRA\_PBIL instead.

### Usage

```
StrLearningPBIL_LDLRA(...)
```

### Arguments

... All arguments passed to [LDLRA\\_PBIL](#)

---

StudentAnalysis *StudentAnalysis*

---

### Description

The StudentAnalysis function returns descriptive statistics for each individual student. Specifically, it provides the number of responses, the number of correct answers, the passage rate, the standardized score, the percentile, and the stanine.

### Usage

```
StudentAnalysis(U, na = NULL, Z = NULL, w = NULL)
```

### Arguments

U	U is a data matrix of the type matrix or data.frame.
na	na argument specifies the numbers or characters to be treated as missing values. <ul style="list-style-type: none"> <li>• ID: Student identifier</li> <li>• NR: Number of responses</li> <li>• NRS: Number-right score (total correct answers)</li> <li>• PR: Passage rate (proportion correct)</li> <li>• SS: Standardized score (z-score)</li> <li>• Percentile: Student's percentile rank</li> <li>• Stanine: Student's stanine score (1-9)</li> </ul>
Z	Z is a missing indicator matrix of the type matrix or data.frame
w	w is item weight vector

**Value**

Returns a data frame containing the following columns for each student:

- ID: Student identifier
- NR: Number of responses
- NRS: Number-right score (total correct answers)
- PR: Passage rate (proportion correct)
- SS: Standardized score (z-score)
- Percentile: Student's percentile rank
- Stanine: Student's stanine score (1-9)

**Examples**

```
# using sample dataset
StudentAnalysis(J15S500)
```

---

 TestFit

*Model Fit Functions for test whole*


---

**Description**

A general function that returns the model fit indices.

**Usage**

```
TestFit(U, Z, ell_A, nparam)
```

**Arguments**

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
Z	Z is a missing indicator matrix of the type matrix or data.frame
ell_A	log likelihood of this model
nparam	number of parameters for this model

**Value**

**model\_log\_like** log likelihood of analysis model  
**bench\_log\_like** log likelihood of benchmark model  
**null\_log\_like** log likelihood of null model  
**model\_Chi\_sq** Chi-Square statistics for analysis model  
**null\_Chi\_sq** Chi-Square statistics for null model  
**model\_df** degrees of freedom of analysis model

**null\_df** degrees of freedom of null model

**NFI** Normed Fit Index. Lager values closer to 1.0 indicate a better fit.

**RFI** Relative Fit Index. Lager values closer to 1.0 indicate a better fit.

**IFI** Incremental Fit Index. Lager values closer to 1.0 indicate a better fit.

**TLI** Tucker-Lewis Index. Lager values closer to 1.0 indicate a better fit.

**CFI** Comparative Fit Index. Lager values closer to 1.0 indicate a better fit.

**RMSEA** Root Mean Square Error of Approximation. Smaller values closer to 0.0 indicate a better fit.

**AIC** Akaike Information Criterion. A lower value indicates a better fit.

**CAIC** Consistent AIC. A lower value indicates a better fit.

**BIC** Bayesian Information Criterion. A lower value indicates a better fit.

---

TestFitSaturated

*Model Fit Functions for saturated model*

---

### Description

A general function that returns the model fit indices.

### Usage

```
TestFitSaturated(U, Z, ell_A, nparam)
```

### Arguments

U	U is either a data class of exametrika, or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
Z	Z is a missing indicator matrix of the type matrix or data.frame
ell_A	log likelihood of this model
nparam	number of parameters for this model

### Value

**model\_log\_like** log likelihood of analysis model

**bench\_log\_like** log likelihood of benchmark model

**null\_log\_like** log likelihood of null model

**model\_Chi\_sq** Chi-Square statistics for analysis model

**null\_Chi\_sq** Chi-Square statistics for null model

**model\_df** degrees of freedom of analysis model

**null\_df** degrees of freedom of null model

**NFI** Normed Fit Index. Lager values closer to 1.0 indicate a better fit.

- RFI** Relative Fit Index. Larger values closer to 1.0 indicate a better fit.
- IFI** Incremental Fit Index. Larger values closer to 1.0 indicate a better fit.
- TLI** Tucker-Lewis Index. Larger values closer to 1.0 indicate a better fit.
- CFI** Comparative Fit Index. Larger values closer to 1.0 indicate a better fit.
- RMSEA** Root Mean Square Error of Approximation. Smaller values closer to 0.0 indicate a better fit.
- AIC** Akaike Information Criterion. A lower value indicates a better fit.
- CAIC** Consistent AIC. A lower value indicates a better fit.
- BIC** Bayesian Information Criterion. A lower value indicates a better fit.

---

TestInformationFunc     *TIF for IRT*

---

### Description

Test Information Function for 4PLM

### Usage

TestInformationFunc(params, theta)

### Arguments

params	parameter matrix
theta	ability parameter

### Value

Returns a numeric vector representing the test information at each ability level theta. The test information is the sum of item information functions for all items in the test:  $I_{test}(\theta) = \sum_{j=1}^n I_j(\theta)$

---

TestResponseFunc     *TRF for IRT*

---

### Description

Calculates the expected score across all items on a test for a given ability level (theta) using Item Response Theory. The Test Response Function (TRF) is essentially the sum of the Item Characteristic Curves (ICCs) for all items in the test.

### Usage

TestResponseFunc(params, theta)

**Arguments**

params	parameter matrix
theta	ability parameter

**Details**

The Test Response Function computes the expected total score for an examinee with a given ability level ( $\theta$ ) across all items in the test. For each item, the function uses the logistic model with parameters  $a$  (discrimination),  $b$  (difficulty),  $c$  (guessing), and  $d$  (upper asymptote).

**Value**

A numeric vector with the same length as  $\theta$ , containing the expected total score for each ability level.

---

TestStatistics	<i>Simple Test Statistics</i>
----------------	-------------------------------

---

**Description**

Calculates descriptive statistics for test scores, providing a comprehensive summary of central tendency, variability, and distribution shape. Different statistics are calculated based on the data type (binary, ordinal, rated, or nominal).

**Usage**

```
TestStatistics(U, na = NULL, Z = NULL, w = NULL)

## Default S3 method:
TestStatistics(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'binary'
TestStatistics(U, na = NULL, Z = NULL, w = NULL)

## S3 method for class 'ordinal'
TestStatistics(U, na = NULL, Z = NULL, w = NULL)
```

**Arguments**

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <code>dataFormat</code> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

**Value**

The returned object depends on the data type:

For binary data, a list of class `c("exametrika", "TestStatistics")` containing:

**TestLength** Length of the test. The number of items included in the test.

**SampleSize** Sample size. The number of rows in the dataset.

**Mean** Average number of correct answers.

**SEofMean** Standard error of mean.

**Variance** Variance of test scores.

**SD** Standard Deviation of test scores.

**Skewness** Skewness of score distribution (measure of asymmetry).

**Kurtosis** Kurtosis of score distribution (measure of tail extremity).

**Min** Minimum score.

**Max** Maximum score.

**Range** Range of scores (Max - Min).

**Q1** First quartile. Same as the 25th percentile.

**Median** Median. Same as the 50th percentile.

**Q3** Third quartile. Same as the 75th percentile.

**IQR** Interquartile range. Calculated by subtracting Q1 from Q3.

**Stanine** Stanine score boundaries, see [stanine](#).

For ordinal and rated data, the function calls [ScoreReport](#) and returns its result. See [ScoreReport](#) for details of the returned object.

For nominal data, an error is returned as this function does not support nominal data.

**Examples**

```
# Basic usage
stats <- TestStatistics(J15S500)
print(stats)

# Extract specific statistics
cat("Mean score:", stats$Mean, "\n")
cat("Standard deviation:", stats$SD, "\n")

# View score distribution summary
summary_stats <- data.frame(
  Min = stats$Min,
  Q1 = stats$Q1,
  Median = stats$Median,
  Mean = stats$Mean,
  Q3 = stats$Q3,
  Max = stats$Max
)
print(summary_stats)
```

```
# Stem-and-leaf plot of score distribution
stem(nrs(dataFormat(J15S500)))
```

---

tetrachoric

*Tetrachoric Correlation*


---

### Description

Tetrachoric Correlation is superior to the phi coefficient as a measure of the relation of an item pair. See Divgi, 1979; Olsson, 1979; Harris, 1988.

### Usage

```
tetrachoric(x, y)
```

### Arguments

x	binary vector x
y	binary vector y

### Value

Returns a single numeric value of class "exametrika" representing the tetrachoric correlation coefficient between the two binary variables. The value ranges from -1 to 1, where:

- 1 indicates perfect positive correlation
- -1 indicates perfect negative correlation
- 0 indicates no correlation

### References

- Divgi, D. R. (1979). Calculation of the tetrachoric correlation coefficient. *Psychometrika*, 44, 169–172.
- Olsson, U. (1979). Maximum likelihood estimation of the polychoric correlation coefficient. *Psychometrika*, 44, 443–460.
- Harris, B. (1988). Tetrachoric correlation coefficient. In L. Kotz, & N. L. Johnson (Eds.), *Encyclopedia of statistical sciences* (Vol. 9, pp. 223–225). Wiley.

---

TetrachoricCorrelationMatrix  
*Tetrachoric Correlation Matrix*

---

## Description

Calculates the matrix of tetrachoric correlations between all pairs of items. Tetrachoric Correlation is superior to the phi coefficient as a measure of the relation of an item pair. This function is applicable only to binary response data.

## Usage

```
TetrachoricCorrelationMatrix(U, na = NULL, Z = NULL, w = NULL)
```

```
## Default S3 method:
```

```
TetrachoricCorrelationMatrix(U, na = NULL, Z = NULL, w = NULL)
```

```
## S3 method for class 'binary'
```

```
TetrachoricCorrelationMatrix(U, na = NULL, Z = NULL, w = NULL)
```

## Arguments

U	Either an object of class "exametrika" or raw data. When raw data is given, it is converted to the exametrika class with the <a href="#">dataFormat</a> function.
na	Values to be treated as missing values.
Z	Missing indicator matrix of type matrix or data.frame. Values of 1 indicate observed responses, while 0 indicates missing data.
w	Item weight vector specifying the relative importance of each item.

## Value

A matrix of tetrachoric correlations with exametrika class. Each element (i,j) represents the tetrachoric correlation between items i and j. The matrix is symmetric with ones on the diagonal.

## Note

This function is implemented using a binary data compatibility wrapper and will raise an error if used with polytomous data.

## Examples

```
# example code  
TetrachoricCorrelationMatrix(J15S500)
```

---

 ThreePLM

*Three-Parameter Logistic Model*


---

**Description**

The three-parameter logistic model is a model where the lower asymptote parameter  $c$  is added to the 2PLM

**Usage**

ThreePLM( $a$ ,  $b$ ,  $c$ ,  $\theta$ )

**Arguments**

$a$	slope parameter
$b$	location parameter
$c$	lower asymptote parameter
$\theta$	ability parameter

**Value**

Returns a numeric vector of probabilities between  $c$  and 1, representing the probability of a correct response given the ability level  $\theta$ . The probability is calculated using the formula:  $P(\theta) = c + \frac{1-c}{1+e^{-a(\theta-b)}}$

---

 TwoPLM

*Two-Parameter Logistic Model*


---

**Description**

The two-parameter logistic model is a classic model that defines the probability of a student with ability  $\theta$  successfully answering item  $j$ , using both a slope parameter and a location parameter.

**Usage**

TwoPLM( $a$ ,  $b$ ,  $\theta$ )

**Arguments**

$a$	slope parameter
$b$	location parameter
$\theta$	ability parameter

**Value**

Returns a numeric vector of probabilities between 0 and 1, representing the probability of a correct response given the ability level theta. The probability is calculated using the formula:  $P(\theta) = \frac{1}{1+e^{-a(\theta-b)}}$

# Index

## \* datasets

- J12S5000, [56](#)
  - J15S3810, [56](#)
  - J15S500, [57](#)
  - J20S400, [57](#)
  - J20S600, [58](#)
  - J21S300, [58](#)
  - J35S500, [59](#)
  - J35S5000, [59](#)
  - J35S515, [60](#)
  - J50S100, [60](#)
  - J5S10, [61](#)
  - J5S1000, [61](#)
- AlphaCoefficient, [4](#)  
AlphaIfDel, [5](#)  
asyprior, [5](#)
- Biclustering, [6](#), [33](#)  
Biclustering\_IRM, [10](#), [33](#), [42](#)  
BINET, [16](#)  
BiserialCorrelation, [19](#)  
BitRespPtn, [20](#)  
BNM, [20](#)  
BNM\_GA, [22](#), [105](#)  
BNM\_PBIL, [24](#), [105](#)
- calcFitIndices, [26](#)  
CCR, [27](#)  
crr, [28](#)  
CSR, [29](#)  
CTT, [30](#), [41](#)
- dataFormat, [7](#), [12](#), [16](#), [21](#), [22](#), [24](#), [27–29](#), [31](#),  
[32](#), [36](#), [40](#), [41](#), [43](#), [44](#), [46](#), [47](#), [49–52](#),  
[54](#), [55](#), [62–65](#), [68](#), [71](#), [74](#), [80](#), [84](#), [86](#),  
[88](#), [90](#), [91](#), [96](#), [100](#), [103](#), [104](#), [107](#),  
[108](#), [110](#), [113](#)
- Dimensionality, [32](#)  
DistractorAnalysis, [33](#)
- GridSearch, [34](#)  
GRM, [36](#)  
grm\_iif, [37](#)  
grm\_prob, [38](#)
- IIF2PLM, [39](#)  
IIF3PLM, [39](#)  
InterItemAnalysis, [40](#)  
IRM, [42](#)  
IRT, [43](#)  
ITBiserial, [44](#)  
ItemEntropy, [45](#)  
ItemFit, [37](#), [43](#), [47](#), [66](#), [81](#)  
ItemInformationFunc, [48](#)  
ItemLift, [48](#)  
ItemOdds, [49](#)  
ItemReport, [50](#)  
ItemStatistics, [52](#)  
ItemThreshold, [53](#)  
ItemTotalCorr, [54](#)
- J12S5000, [56](#)  
J15S3810, [56](#)  
J15S500, [57](#)  
J20S400, [57](#)  
J20S600, [58](#)  
J21S300, [58](#)  
J35S500, [59](#)  
J35S5000, [59](#)  
J35S515, [60](#)  
J50S100, [60](#)  
J5S10, [61](#)  
J5S1000, [61](#)  
JCRR, [62](#)  
JointSampleSize, [63](#)  
JSR, [64](#)
- LCA, [64](#)  
LD\_param\_est, [76](#)  
LDB, [67](#)

LDLRA, 70  
LDLRA\_PBIL, 73, 106  
LogisticModel, 76  
longdataFormat, 77  
LRA, 33, 78

maxParents\_penalty, 83  
MutualInformation, 84

nrs, 85

objective\_function\_IRT, 87  
OmegaCoefficient, 87

passage, 88  
percentile, 89  
PhiCoefficient, 90  
plot.DistractorAnalysis  
    (DistractorAnalysis), 33  
plot.exametrika, 13, 17, 69, 82, 91  
polychoric, 95  
PolychoricCorrelationMatrix, 96  
polyserial, 97  
print.DistractorAnalysis  
    (DistractorAnalysis), 33  
print.exametrika, 98  
PSD\_item\_params, 99

RaschModel, 100

ScoreReport, 100, 111  
slopeprior, 101  
softmax, 102  
sscore, 102  
stanine, 103, 111  
StrLearningGA\_BNM, 105  
StrLearningPBIL\_BNM, 105  
StrLearningPBIL\_LDLRA, 106  
StudentAnalysis, 106

TestFit, 13, 18, 21, 23, 25, 37, 43, 66, 69, 72,  
    75, 81, 107  
TestFitSaturated, 108  
TestInformationFunc, 109  
TestResponseFunc, 109  
TestStatistics, 110  
tetrachoric, 112  
TetrachoricCorrelationMatrix, 113  
ThreePLM, 114  
TwoPLM, 114