Package ‘iotarelr’

February 28, 2023

Type Package

Title Iota Inter Coder Reliability for Content Analysis

Version 0.1.4

Description Routines and tools for assessing the quality of content analysis on the basis of the Iota Reliability Concept. The concept is inspired by item response theory and can be applied to any kind of content analysis which uses a standardized coding scheme and discrete categories. It is also applicable for content analysis conducted by artificial intelligence. The package provides reliability measures for a complete scale as well as for every single category. Analysis of subgroup-invariance and error corrections are implemented. This information can support the development process of a coding scheme and allows a detailed inspection of the quality of the generated data. Equations and formulas working in this package are part of Berding et al. (2022)<doi:10.3389/feduc.2022.818365> and Berding and Pargmann (2022) <doi:10.30819/5581>.

License GPL-3

URL https://fberding.github.io/iotarelr/

BugReports https://github.com/FBerding/iotarelr/issues

Depends R (>= 3.5.0), ggplot2, ggalluvial,
Imports gridExtra, methods, Rcpp, rlang, stats
Suggests knitr, rmarkdown, testthat (>= 3.0.0)

LinkingTo Rcpp

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.1

NeedsCompilation yes

Author Berding Florian [aut, cre] (<https://orcid.org/0000-0002-3593-1695>), Pargmann Julia [ctb]
check_conformity_c

Check assumptions of weak superiority

Description

This function tests if the probabilities within the Assignment Error Matrix are in line with the assumption of weak superiority.

Usage

check_conformity_c(aem)

Arguments

aem matrix of probabilities
check_dgf

Value

Returns the number of violations of the assumption of weak superiority. 0 if the assumptions are fulfilled.

References


---

check_dgf

Check for Different Guidance Functioning (DGF)

---

Description

Function for checking if the coding scheme is the same for different sub-groups.

Usage

```
check_dgf(
  data,
  splitcr,
  random_starts = 300,
  max_iterations = 5000,
  cr_rel_change = 1e-12,
  con_step_size = 1e-04,
  con_random_starts = 10,
  con_max_iterations = 5000,
  con_rel_convergence = 1e-12,
  b_min = 0.01,
  trace = FALSE,
  con_trace = FALSE,
  fast = TRUE
)
```

Arguments

data Data for which the elements should be estimated. Data must be an object of type data.frame or matrix with cases in the rows and raters in the columns. Please note that no additional variables are allowed in this object.

splitcr Vector containing the assignments of coding units to groups. The vector must have the same length as the number of rows of object data.

random_starts An integer for the number of random starts for the EM algorithm.

max_iterations An integer for the maximum number of iterations within the EM algorithm.

cr_rel_change Positive numeric value for defining the convergence of the EM algorithm.

con_step_size Double for specifying the size for increasing or decreasing the probabilities during the conditioning stage of estimation. This value should not be less than 1e-3.
check_new_rater

**check_new_rater**

Function for estimating the reliability of codings for a new rater based on Iota 2

**Usage**

```r
check_new_rater(
  true_values,
  assigned_values,
  con_step_size = 1e-04,
  con_random_starts = 5,
  con_max_iterations = 5000,
  con_rel_convergence = 1e-12,
  con_trace = FALSE,
)
```

**con_random_starts**

Integer for the number of random starts within the condition stage.

**con_max_iterations**

Integer for the maximum number of iterations during the condition stage.

**con_rel_convergence**

Double for determining the convergence criterion during condition stage. The algorithm stops if the relative change is smaller than this criterion.

**b_min**

Value ranging between 0 and 1 determining the minimal size of the categories for checking if boundary values occurred. The algorithm tries to select solutions that are not considered to be boundary values.

**trace**

TRUE for printing progress information on the console. FALSE if this information is not to be printed.

**con_trace**

TRUE for printing progress information on the console during estimations in the condition stage. FALSE if this information is not to be printed.

**fast**

Bool If TRUE a fast estimation is applied during the condition stage. This option ignores all parameters beginning with "con_". If FALSE the estimation described in Berding and Pargmann (2022) is used. Default is TRUE.

**Value**

Returns an object of class `iotarelr_iota2_dif`. For each group, the results of the estimation are saved separately. The structure within each group is similar to the results from `compute_iota2()`. Please check that documentation.

**References**

check_new_rater

```r
fast = TRUE,
free_aem = FALSE
)
```

**Arguments**

- **true_values** Vector containing the true categories of the coding units. Vector must have the same length as `assigned_values`.
- **assigned_values** Vector containing the assigned categories of the coding units. Missing values are currently not supported and have to be omitted from the vector. Vector must have the same length as `true_values`.
- **con_step_size** Double for specifying the size for increasing or decreasing the probabilities during the conditioning stage of estimation. This value should not be less than 1e-3.
- **con_random_starts** Integer for the number of random starts within the condition stage.
- **con_max_iterations** Integer for the maximum number of iterations during the conditioning stage.
- **con_rel_convergence** Double for determining the convergence criterion during the conditioning stage. The algorithm stops if the relative change is smaller than this criterion.
- **con_trace** TRUE for printing progress information on the console during estimations in the conditioning stage. FALSE if you do not want to have this information printed.
- **fast** Bool If TRUE a fast estimation is applied during the condition stage. This option ignores all parameters beginning with "con_". If FALSE the estimation described in Berding and Pargmann (2022) is used. Default is TRUE.
- **free_aem** Bool If TRUE the Assignment Error Matrix is estimated in a way ensuring conformity with the assumption of weak superiority. If FALSE the Assignment Error Matrix is freely estimated. TRUE is default.

**Value**

Returns a list with the following three components: The first component `estimates_categorical_level` comprises all elements that describe the ratings on a categorical level. The elements are sub-divided into raw estimates and chance-corrected estimates.

- **raw_estimates**
  - **alpha_reliability**: A vector containing the Alpha Reliabilities for each category. These values represent probabilities.
  - **beta_reliability**: A vector containing the Beta Reliabilities for each category. These values represent probabilities.
  - **assignment_error_matrix**: An Assignment Error Matrix containing the conditional probabilities for assigning a unit of category i to categories 1 to n.
  - **iota**: A vector containing the Iota values for each category.

- **elements_chance_corrected**
  - **alpha_reliability**: A vector containing the chance-corrected Alpha Reliabilities for each category.
  - **beta_reliability**: A vector containing the chance-corrected Beta Reliabilities for each category.
The second component estimates_scale_level contains elements to describe the quality of the ratings on a scale level. It contains the following elements:

- **iota_index**: The Iota Index representing the reliability on a scale level.
- **iota_index_d4**: The Static Iota Index, which is a transformation of the original Iota Index, in order to consider the uncertainty of estimation.
- **iota_index_dyn2**: The Dynamic Iota Index, which is a transformation of the original Iota Index, in order to consider the uncertainty of estimation.

The third component information contains important information regarding the parameter estimation. It comprises the following elements:

- **log_likelihood**: Log-likelihood of the best solution.
- **convergence**: If estimation converged 0, otherwise 1.
- **est_true_cat_sizes**: Estimated categorical sizes. This is the estimated amount of the categories.
- **conformity**: 0 if the solution is in line with assumptions of weak superiority. A number greater 0 indicates the number of violations of the assumption of weak superiority.
- **random_starts**: Numer of random starts for the EM algorithm.
- **boundaries**: False if the best solution does not contain boundary values. True if the best solution does contain boundary values.
- **p_boundaries**: Percentage of solutions with boundary values during estimation.
- **call**: Name of the function that created the object.
- **n_rater**: Number of raters.
- **n_cunits**: Number of coding units.

**Note**

The returned object contains further slots since the returned object is of class `iotarelr_iota2`. These slots are empty because they are not part of the estimation within this function.

Please do not use the measures on the scale level if the Assignment Error Matrix was freely estimated since this kind of matrix is not conceptualized for comparing the coding process with random guessing.

**References**

compute_iota1

---

**compute_iota1**  Computes Iota and its elements in version 1

---

**Description**

Computes all elements of the Iota Reliability Concept

**Usage**

```r
compute_iota1(data)
```

**Arguments**

data  Data for which the elements should be estimated. Data must be an object of type `data.frame` or `matrix` with cases in the rows and raters in the columns.

**Value**

A list with the following components

- **alpha**: A vector containing the chance-corrected Alpha Reliabilities for every category.
- **beta**: A vector containing the chance-corrected Beta Reliabilities for every category.
- **iota**: A vector containing the Iota values for every category.
- **assignment_error_matrix**: A matrix with the conditional probabilities for every category. The rows refer to the true categories and the columns refer to the assigned categories. The elements on the diagonal represent the alpha errors of that category. The other elements in each row represent the conditioned probabilities that a coding unit is wrongly assigned to another category.
- **average_iota**: A numeric value ranging between 0 and 1, representing the Average Iota values on a categorical level. It describes the reliability of the whole scale.

**References**

**compute_iota2**

*Computes Iota and its elements in version 2*

**Description**

Fits a model of Iota2 to the data

**Usage**

```r
compute_iota2(
  data,
  random_starts = 10,
  max_iterations = 5000,
  cr_rel_change = 1e-12,
  con_step_size = 1e-04,
  con_rel_convergence = 1e-12,
  con_max_iterations = 5000,
  con_random_starts = 5,
  b_min = 0.01,
  fast = TRUE,
  trace = TRUE,
  con_trace = FALSE
)
```

**Arguments**

- **data**
  Data for which the elements should be estimated. Data must be an object of type `data.frame` or `matrix` with cases in the rows and raters in the columns.

- **random_starts**
  An integer for the number of random starts for the EM algorithm.

- **max_iterations**
  An integer for the maximum number of iterations within the EM algorithm.

- **cr_rel_change**
  Positive numeric value for defining the convergence of the EM algorithm.

- **con_step_size**
  Double for specifying the size for increasing or decreasing the probabilities during the conditioning stage of estimation. This value should not be less than 1e-3.

- **con_rel_convergence**
  Double for determining the convergence criterion during the conditioning stage. The algorithm stops if the relative change is smaller than this criterion.

- **con_max_iterations**
  Integer for the maximum number of iterations during the conditioning stage.

- **con_random_starts**
  Integer for the number of random starts within the conditioning stage.

- **b_min**
  Value ranging between 0 and 1, determining the minimal size of the categories for checking if boundary values occurred. The algorithm tries to select solutions that are not considered to be boundary values.
**compute_iota2**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fast</td>
<td>Bool If TRUE a fast estimation is applied during the condition stage. This option ignores all parameters beginning with &quot;con_&quot;. If FALSE the estimation described in Berding and Pargmann (2022) is used. Default is TRUE.</td>
</tr>
<tr>
<td>trace</td>
<td>TRUE for printing progress information on the console. FALSE if this information is not to be printed.</td>
</tr>
<tr>
<td>con_trace</td>
<td>TRUE for printing progress information on the console during estimations in the conditioning stage. FALSE if this information is not to be printed.</td>
</tr>
</tbody>
</table>

**Value**

Returns a list with the following three components: The first component `estimates_categorical_level` comprises all elements that describe the ratings on a categorical level. The elements are sub-divided into raw estimates and chance-corrected estimates.

- **raw_estimates**
  - **alpha_reliability**: A vector containing the Alpha Reliabilities for each category. These values represent probabilities.
  - **beta_reliability**: A vector containing the Beta Reliabilities for each category. These values represent probabilities.
  - **assignment_error_matrix**: Assignment Error Matrix containing the conditional probabilities for assigning a unit of category i to categories 1 to n.
  - **iota**: A vector containing the Iota values for each category.
  - **iota_error_1**: A vector containing the Iota Error Type I values for each category.
  - **iota_error_2**: A vector containing the Iota Error Type II values for each category.

- **elements_chance_corrected**
  - **alpha_reliability**: A vector containing the chance-corrected Alpha Reliabilities for each category.
  - **beta_reliability**: A vector containing the chance-corrected Beta Reliabilities for each category.

The second component `estimates_scale_level` contains elements for describing the quality of the ratings on a scale level. It comprises the following elements:

- **iota_index**: The Iota Index, representing the reliability on a scale level.
- **iota_index_d4**: The Static Iota Index, which is a transformation of the original Iota Index, in order to consider the uncertainty of estimation.
- **iota_index_dyn2**: The Dynamic Iota Index, which is a transformation of the original Iota Index, in order to consider the uncertainty of estimation.

The third component `information` contains important information regarding the parameter estimation. It comprises the following elements:

- **log_likelihood**: Log-likelihood of the best solution.
- **convergence**: If estimation converged 0, otherwise 1.
- **est_true_cat_sizes**: Estimated categorical sizes. This is the estimated amount of the categories.
- **conformity**: 0 if the solution is in line with assumptions of weak superiority. A number greater 0 indicates the number of violations of the assumption of weak superiority.
- **random_starts**: Number of random starts for the EM algorithm.
- **boundaries**: False if the best solution does not contain boundary values. True if the best solution does contain boundary values.
• p_boundaries: Percentage of solutions with boundary values during the estimation.
• call: Name of the function that created the object.
• n_rater: Number of raters.
• n_cunits: Number of coding units.

References

---

**EM_algo_c**

*Parameter estimation via EM Algorithm with Condition Stage*

---

**Description**

Function written in C++ for estimating the parameters of the model via Expectation Maximization (EM Algorithm).

**Usage**

```c++
EM_algo_c(
    obs_pattern_shape,
    obs_pattern_frq,
    obs_internal_count,
    categorical_levels,
    random_starts,
    max_iterations,
    rel_convergence,
    con_step_size,
    con_random_starts,
    con_max_iterations,
    con_rel_convergence,
    fast,
    trace,
    con_trace
)
```

**Arguments**

- `obs_pattern_shape`
  Matrix containing the unique patterns found in the data. Ideally this matrix is generated by the function `get_patterns()`.

- `obs_pattern_frq`
  Vector containing the frequencies of the patterns. Ideally it is generated by the function `get_patterns()`.
obs_internal_count
Matrix containing the relative frequencies of each category within each pattern. Ideally this matrix is generated by the function get_patterns().

categorical_levels
Vector containing all possible categories of the content analysis.

random_starts
Integer for determining how often the algorithm should restart with randomly chosen values for the Assignment Error Matrix and the categorical sizes.

max_iterations
Integer for determining the maximum number of iterations for each random start.

rel_convergence
Double for determining the convergence criterion. The algorithm stops if the relative change is smaller than this criterion.

con_step_size
Double for specifying the size for increasing or decreasing the probabilities during the condition stage of estimation. This value should not be less than 1e-3.

con_random_starts
Integer for the number of random starts within the condition stage.

con_max_iterations
Integer for the maximum number of iterations during the condition stage.

con_rel_convergence
Double for determining the convergence criterion during condition stage. The algorithm stops if the relative change is smaller than this criterion.

fast
Bool If TRUE a fast estimation is applied during the condition stage. This option ignores all parameters beginning with "con_." If FALSE the estimation described in Berding and Pargmann (2022) is used. Default is TRUE.

trace
TRUE for printing progress information on the console. FALSE if this information should not be printed.

con_trace
TRUE for printing progress information on the console during estimations in the condition stage. FALSE if this information should not be printed.

Value
Function returns a list with the estimated parameter sets for every random start. Every parameter set contains the following components:

log_likelihood
Log likelihood of the estimated solution.

aem
Estimated Assignment Error Matrix (aem). The rows represent the true categories while the columns stand for the assigned categories. The cells describe the probability that a coding unit of category i is assigned to category j.

categorical_sizes
Vector of estimated sizes for each category.

convergence
If the algorithm converged within the iteration limit TRUE. FALSE in every other case.

iteration
Number of iterations when the algorithm was terminated.
References


est_con_multinominal_c

Estimating log likelihood in Condition Stage

Description

Function written in C++ estimating the log likelihood of a given parameter set during the condition stage.

Usage

est_con_multinominal_c(
  observations,
  anchor,
  max_iter = 500000L,
  step_size = 1e-04,
  cr_rel_change = 1e-12,
  n_random_starts = 10L,
  fast = TRUE,
  trace = FALSE
)

Arguments

- observations: NumericVector containing the frequency of the categories.
- anchor: Integer ranging between 1 and the number of categories. Anchor defines the reference category. That is the category with the highest probability according to the assumption of weak superiority.
- max_iter: Integer specifying the maximal number of iterations for each random start.
- step_size: Double for specifying the size for increasing or decreasing the probabilities during the estimation. This value should not be less than 1e-3.
- cr_rel_change: Double for defining when the estimation should stop. That is, if the change in log-likelihood is smaller as this value the estimation stops.
- n_random_starts: Integer for the number of random start.
- fast: Bool If TRUE a fast estimation is applied. This option ignored all other parameters. If FALSE the estimation described in Berding and Pargmann (2022) is used. Default is TRUE.
- trace: Bool TRUE if information about the progress of estimation should be printed to the console. FALSE if not desired.
**est_expected_categories**

**Value**

Returns the log likelihood as a single numeric value.

**References**


---

**est_expected_categories**

*Estimate Expected Categories*

---

**Description**

Function for estimating the expected category of coding units.

**Usage**

```
est_expected_categories(data, aem)
```

**Arguments**

- **data**
  - Matrix which contains the codings for every coding unit. The coding units must be in the rows and the raters must be in the columns. At least two raters are necessary.

- **aem**
  - Assignment Error Matrix based on the second generation of the Iota Concept (Iota2).

**Value**

Returns a matrix with the original data, the conditioned probability of each true category, and the expected category for every coding unit.

**References**

fct_log_likelihood_c  Estimating log-likelihood

Description

Function written in C++ estimating the log likelihood of a given parameter set.

Usage

fct_log_likelihood_c(
  categorial_sizes,  
  aem,  
  obs_pattern_shape,  
  obs_pattern_frq,  
  categorical_levels
)

Arguments

- **categorial_sizes**  
  Vector containing the sizes of the different categories. That is amount of a category on all cases.

- **aem**  
  Matrix in aem form. This matrix reports the true category in the rows and the assigned categories in the columns. The cells represent the probabilities that a coding unit of category i is assigned to category j.

- **obs_pattern_shape**  
  Matrix containing the unique patterns found in the data. Ideally this matrix is generated by the function get_patterns().

- **obs_pattern_frq**  
  Vector containing the frequencies of the patterns. Ideally it is generated by the function get_patterns().

- **categorical_levels**  
  Vector containing all possible categories of the content analysis.

Value

Returns the log likelihood as a single numeric value.

References

**get_consequences**  

**Get Consequences**

**Description**

Function estimating the consequences of reliability for subsequent analysis.

**Usage**

```r
get_consequences(
  measure_typ = "dynamic_iota_index",
  measure_1_val,
  measure_2_val = NULL,
  level = 0.95,
  strength = NULL,
  data_type,
  sample_size
)
```

**Arguments**

- **measure_typ**  
  Type of measure used for estimation. Set "iota_index" for the original Iota Index, "static_iota_index" for the static transformation of the Iota Index with d=4 or "dynamic_iota_index" for the dynamic transformation of the Iota Index with d=2.

- **measure_1_val**  
  Reliability value for the independent variable.

- **measure_2_val**  
  Reliability value for the dependent variable. If not set, the function uses the same value as for the independent variable.

- **level**  
  Level of certainty for calculating the prediction intervals.

- **strength**  
  True strength of the relationship between the independent and dependent variable. Possible values are "no", "weak", "medium" and "strong". If no value is supplied, a strong relationship is assumed for deviation and a weak relationship for all others. They represent the most demanding situations for the reliability.

- **data_type**  
  Type of data. Possible values are "nominal" or "ordinal".

- **sample_size**  
  Size of the sample in the study.

**Value**

Returns a data.frame which contains the prediction intervals for the deviation between true and estimated sample association/correlation, risk of Type I errors and chance to correctly classify the effect size. Additionally, the probability is estimated so that the statistics of the sample deviate from an error free sample with no or only a weak effect.
Note

The classification of effect sizes uses the work of Cohen (1988), who differentiates effect sizes by their relevance for practice.

For nominal data, all statistics refer to Cramer’s V. For ordinal data, all statistics refer to Kendall’s Tau.

The models for calculating the consequences are taken from Berding and Pargmann (2022).

References


get_iota2_measures  Get Iota2 Measures

Description

Function for calculating the elements of the Iota Concept 2

Usage

get_iota2_measures(aem, categorical_sizes, categorical_levels)

Arguments

aem  Assignment Error Matrix.
categorical_sizes  Probabilities for the different categories to occur.
categorical_levels  Vector containing all possible categories of the content analysis.

Value

Returns a list of all measures belonging to the Iota Concept of the second generation. The first component estimates_categorical_level comprises all elements that describe the ratings on a categorical level. The elements are sub-divided into raw estimates and chance-corrected estimates.

raw_estimates  • iota: A vector containing the Iota values for each category.
  • iota_error_1: A vector containing the Iota Error Type I values for each category.
  • iota_error_2: A vector containing the Iota Error Type II values for each category.
  • alpha_reliability: A vector containing the Alpha Reliabilities for each category. These values represent probabilities.
  • beta_reliability: A vector containing the Beta Reliabilities for each category. These values represent probabilities.
get_patterns

- assignment_error_matrix: Assignment Error Matrix containing the conditional probabilities for assigning a unit of category i to categories 1 to n.

- alpha_reliability: A vector containing the chance-corrected Alpha Reliabilities for each category.

- beta_reliability: A vector containing the chance-corrected Beta Reliabilities for each category.

The second component estimates_scale_level contains elements for describing the quality of the ratings on a scale level. It comprises the following elements:

- iota_index: The Iota Index, representing the reliability on a scale level.

- iota_index_d4: The Static Iota Index, which is a transformation of the original Iota Index, in order to consider the uncertainty of estimation.

- iota_index_dyn2: The Dynamic Iota Index, which is a transformation of the original Iota Index, in order to consider the uncertainty of estimation.

References


---

get_patterns  Get patterns

Description

Auxiliary function written in R for providing the necessary information about the patterns generated by raters. This function produces the input for the EM-algorithm.

Usage

get_patterns(data, categorical_levels)

Arguments

data  Matrix or data.frame containing the ratings. The cases are in the rows and the raters are in the columns. Characters in the cells are supported. At least two raters are necessary.

categorical_levels  Vector containing all possible categories of the content analysis.
**Value**

Function returns a list with the following components:

- **n** Integer representing the number of different patterns in the data.
- **shape** Matrix containing all unique patterns in the data. Cells of the matrix are characters.
- **freq** Vector containing the frequencies of the patterns.
- **count** Matrix containing the relative frequencies of the categories within each pattern. The number of rows equals the number of patterns. The number of columns equals the number of categories.

---

**get_random_start_values_class_sizes**

*Generating randomly chosen probabilities for categorical sizes*

---

**Description**

Function written in C++ for generating a set of randomly chosen probabilities describing the size of the different classes. The probabilities describe the relative frequencies of the categories in the data.

**Usage**

```r
get_random_start_values_class_sizes(n_categories)
```

**Arguments**

- **n_categories** Integer for the number of categories in the data. Must be at least 2.

**Value**

Returns a vector of randomly chosen categorical sizes.

---

**get_random_start_values_p**

*Generating randomly chosen probabilities for Assignment Error Matrix*

---

**Description**

Function written in C++ for generating a set of randomly chosen probabilities for the Assignment Error Matrix.

**Usage**

```r
get_random_start_values_p(n_categories)
```
**get_summary**

**Arguments**

- n_categories: Integer for the number of categories in the data. Must be at least 2.

**Value**

Returns a matrix for Assignment Error Matrix (AEM) with randomly generated probabilities. The generated probabilities are in line with the assumption of weak superiority.

---

**Description**

Function for creating a short summary of the estimated Iota components.

**Usage**

```r
get_summary(object)
```

**Arguments**

- object: An object of class `iotarelr_iota2` created by `compute_iota2`, `check_new_rater`, or `check_dgf`.

**Value**

Prints central statistics of the estimated model.

---

**grad_ll**

**Gradient for Log Likelihood in Condition Stage**

**Description**

Function written in C++ estimating the gradient of the log likelihood function for a given parameter set and given observations.

**Usage**

```r
grad_ll(param_values, observations)
```

**Arguments**

- param_values: NumericVector containing the probabilities of a multinomial distribution. The length of this factor is the number of categories - 1 since it contains only the parameters to be estimated.
- observations: NumericVector containing the number of observations for each category of the multinominal distribution. The length of this vector equals the number of categories.
Value

Returns the gradient as a NumericVector.

References


---

iotarelr_new_rater  Sample Vector

Description

A vector containing the ratings of a new rater. The data is not real and is only created for illustration purposes.

Usage

iotarelr_new_rater

Format

A vector with the length of 318.

---

iotarelr_written_exams  Example Data Set

Description

A data set containing the ratings of three coders for written exams. It also contains the gender of the people who took the exam. The data is not real and is only created for illustrating purposes.

Usage

iotarelr_written_exams

Format

A data frame with 318 rows and 4 variables:

- **Coder A**  Ratings of coder A.
- **Coder B**  Ratings of coder B.
- **Coder C**  Ratings of coder C.
- **Sex**  Referring to the biological aspects of an individual.
**log_likelihood_multi_c**

*Estimating log-likelihood in Condition Stage*

---

**Description**

Function written in C++ estimating the log likelihood of a given parameter set during the condition stage.

**Usage**

$log\_likelihood\_multi\_c$(probabilities, observations)

**Arguments**

- **probabilities** NumericVector containing the probabilities of a multinominal distribution. In the context of Iota Reliability this refers to a specific row of the Assignment Error Matrix.
- **observations** NumericVector containing the number of observations for each category of the multinominal distribution.

**Value**

Returns the log likelihood as a single numeric value.

**References**


---

**plot_iota**

*Plot Iota2*

---

**Description**

Function for creating a plot object that can be plotted via ggplot2.

**Usage**

`plot_iota`

object,

- `xlab = "Amount on all cases"`,
- `ylab = "Categories"`,
- `liota = "Assignment of the true category (Iota)"`,
- `lcase2 = "Assignment to the false category"`,

---
lcase3 = "Assignment from the false true category",
lscale_quality = "Scale Quality",
lscale_cat = c("insufficient", "minimum", "satisfactory", "good", "excellent"),
number_size = 6,
key_size = 0.5,
text_size = 10,
scale = "none"
)

Arguments

- **object**
  Estimates of Iota 2 created with `compute_iota2()`, `check_dgf()` or `check_new_rater()`.

- **xlab**
  Character passed to `xlab()` from `scale_fill_manual()`. Label of the x-axis.

- **ylab**
  Character passed to `ylab()` from `scale_fill_manual()`. Label of the y-axis.

- **liota**
  Character passed to `labels()` from `scale_fill_manual()`. Label for Iota.Amount of cases that are assigned to the correct category.

- **lcase2**
  Character passed to `labels()` from `scale_fill_manual()`. Label for the amount of cases that are assigned to a false category.

- **lcase3**
  Character passed to `labels()` from `scale_fill_manual()`. Label for the amount of cases that are assigned from a false category.

- **lscale_quality**
  Character passed to `scale_fill_manual()` determining the title for the quality of a scale. Only used in conjunction with `scale`.

- **lscale_cat**
  Vector of strings with length 5. This vector contains the labels for each category of quality for the scale.

- **number_size**
  Double passed to `geom_text()` determining the size of the numbers within the plot.

- **key_size**
  Double passed to `theme()` determining the size of the legend keys.

- **text_size**
  Double passed to `theme()` determining the size of the text within the legend.

- **scale**
  String for requesting an additional plot of reliability on the scale level. If `scale="dynamic_iota_index"` Dynamic Iota Index is used. If `scale="static_iota_index"` Static Iota Index is used. If `scale="none"` no additional plot is created.

Value

Function returns an object of class `gg`, `ggplot` illustrating how the data of the different categories influence each other.

Note

An example for interpreting the plot can be found in the vignette `Get started` or via `vignette("iotarelr", package = "iotarelr")`.

References

plot_iota2_alluvial  

Plot of the Coding Stream

Description

Function for creating an alluvial plot that can be plotted via ggplot2.

Usage

plot_iota2_alluvial(
  object,
  label_titel = "Coding Stream from True to Assigned Categories",
  label_prefix_true = "true",
  label_prefix_assigned = "labeled as",
  label_legend_title = "True Categories",
  label_true_category = "True Category",
  label_assigned_category = "Assigned Category",
  label_y_axis = "Relative Frequencies",
  label_categories_size = 3,
  key_size = 0.5,
  text_size = 10
)

Arguments

object  
Estimates of Iota 2 created with compute_iota2(), check_new_rater() or with check_dgf(). Please note that the object created by check_dgf() cannot be passed directly. Only the elements of the corresponding list are compatible.

label_titel  
Character containing the title of the plot.

label_prefix_true  
Character representing the prefix for tagging the true categories. Character is applied to every category.

label_prefix_assigned  
Character representing the prefix for tagging the assigned categories. Character is applied to every category.

label_legend_title  
Character containing the title of the legend.

label_true_category  
Character describing the stratum of true categories.

label_assigned_category  
Character describing the stratum of assigned categories.

label_y_axis  
Character. Label of the y-axis.

label_categories_size  
double determining the size of the label for each true and assigned category within the plot.
key_size double determining the size of the legend.
text_size double determining the size of the text within the legend.

Value

Returns an object of class gg and ggplot which can be shown with plot().

Note

An example for interpreting the plot can be found in the vignette Get started or via vignette("iotarelr", package = "iotarelr").
Index

* datasets
  - iotarel_r_new_rater, 20
  - iotarel_r_written_exams, 20

check_conformity_c, 2
check_dgf, 3, 19
check_new_rater, 4, 19
compute_iota1, 7
compute_iota2, 8, 19

EM_algo_c, 10
est_con_multinominal_c, 12
est_expected_categories, 13

fct_log_likelihood_c, 14
get_consequences, 15
get_iota2_measures, 16
get_patterns, 17
get_random_start_values_class_sizes,
  18
get_random_start_values_p, 18
get_summary, 19
ggplot2, 21, 23
grad_ll, 19

iotarel_r_new_rater, 20
iotarel_r_written_exams, 20

log_likelihood_multi_c, 21

plot_iota, 21
plot_iota2_alluvial, 23