Package 'logmult'

October 13, 2022

Type Package

Title Log-Multiplicative Models, Including Association Models

Version 0.7.4

Date 2022-02-22

Imports stats, utils, graphics, grDevices, qvcalc

Depends gnm (>= 1.0-5)

Suggests survey (>= 3.34), boot, ellipse, parallel, knitr, rmarkdown

Description Functions to fit log-multiplicative models using 'gnm', with support for convenient printing, plots, and jackknife/bootstrap standard errors. For complex survey data, models can be fitted from design objects from the 'survey' package. Currently supported models include UNIDIFF (Erikson & Goldthorpe, 1992), a.k.a. log-multiplicative layer effect model (Xie, 1992) <doi:10.2307/2096242>, and several association models: Goodman (1979) <doi:10.2307/2286971> row-column association models of the RC(M) and RC(M)-L families with one or several dimensions; two skew-symmetric association models proposed by Yamaguchi (1990) <doi:10.2307/271086> and by van der Heijden & Mooijaart (1995) <doi:10.1177/0049124195024001002> Functions allow computing the intrinsic association coefficient (see Bouchet-Valat (2022) <doi:10.1177/0049124119852389>) and the Altham (1970) index <doi:10.1111/j.2517-6161.1970.tb00816.x>, including via the Bayes shrinkage estimator proposed by Zhou (2015) <doi:10.1177/0081175015570097>; and the RAS/IPF/Deming-Stephan algorithm.

License GPL (>= 2)

URL https://github.com/nalimilan/logmult

BugReports https://github.com/nalimilan/logmult/issues

VignetteBuilder knitr

NeedsCompilation no

Author Milan Bouchet-Valat [aut, cre],
Heather Turner [ctb],
Michael Friendly [ctb],
Jim Lemon [cph],
Gabor Csardi [cph]

Maintainer Milan Bouchet-Valat <nalimilan@club.fr>

Repository CRAN

Date/Publication 2022-02-23 13:50:05 UTC

R topics documented:

anoas *Analysis of Association Functions*

Description

These functions allow performing in a straightforward and efficient way an analysis of association (ANOAS) consisting of successive RC(M) or RC(M)-L models from 1 to N dimensions. They fit the models efficiently by using scores from the previous model as starting values for the next one.

Usage

```
anoas(tab, nd = 3, symmetric = FALSE, diagonal = FALSE, ...)

anoasL(tab, nd = 3,
       layer.effect = c("homogeneous.scores", "heterogeneous", "none"),
       symmetric = FALSE,
       diagonal = c("none", "heterogeneous", "homogeneous"), ...)
```

Arguments

tab	a two-way table, or an object (such as a matrix) that can be coerced into a table; if present, dimensions above two will be collapsed as appropriate.
nd	the number of dimensions to include in the most complex model. Cannot exceed `min(nrow(tab) - 1, ncol(tab) - 1)` if `symmetric` is `FALSE` (saturated model), and twice this threshold otherwise (quasi-symmetry model).
symmetric	See `rc` or `rcL`
layer.effect	See `rcL`.
diagonal	See `rc` or `rcL`.
...	more arguments to be passed to `rc` or `rcL`.

Details

Contrary to most analyses of association in the literature, this function currently does not fit uniform association model ("U"), nor separate models with only row and column association ("R" and "C" models), nor log-linear row and column association models.

Currently, no significance test is performed on the models. Please note that it is not correct to test the one-dimension association model against the independence model.

Value

A `list` of `gnm` objects. The first element is the independence model, the remaining ones are `rc` (for `anoas`) or `rcL` (for `anoasL`) objects with dimensions from 1 to `nd`.

Author(s)

Milan Bouchet-Valat

References

Wong, R.S-K. (2010). Association models. SAGE: Quantitative Applications in the Social Sciences.

See Also

rc, rcL, gnm

Examples

```
## Wong (2010), Table 2.6
data(gss8590)

# The table used in Wong (2010) is not perfectly consistent
# with that of Wong (2001)
tab <- margin.table(gss8590[,,c(2,4)], 1:2)
tab[2,4] <- 49

# Results correspond to lines 1, 6 and 11
results <- anoas(tab, nd=2)
results
```

---

assoc *Identifying Scores from Association Models*

Description

Identify log-multiplicative association scores from over-parameterized gnm models.

Usage

```
## S3 method for class 'rc'
assoc(model, weighting = c("marginal", "uniform", "none"),
                  rowsup = NULL, colsup = NULL, ...)

## S3 method for class 'rc.symm'
assoc(model, weighting = c("marginal", "uniform", "none"),
                       rowsup = NULL, colsup = NULL, ...)

## S3 method for class 'hmskew'
assoc(model, weighting = c("marginal", "uniform", "none"),
                      rowsup = NULL, colsup = NULL, ...)

## S3 method for class 'yrcskew'
assoc(model, weighting = c("marginal", "uniform", "none"), ...)
```

```
## S3 method for class 'rcL'
assoc(model, weighting = c("marginal", "uniform", "none"), ...)

## S3 method for class 'rcL.symm'
assoc(model, weighting = c("marginal", "uniform", "none"), ...)

## S3 method for class 'rcL.trans'
assoc(model, weighting = c("marginal", "uniform", "none"), ...)

## S3 method for class 'hmskewL'
assoc(model, weighting = c("marginal", "uniform", "none"), ...)

## S3 method for class 'rcL.trans.symm'
assoc(model, weighting = c("marginal", "uniform", "none"), ...)
```

Arguments

model	a gnm object, usually obtained using rc, hmskew, yrcskew, rcL, or rcL.trans, but not necessarily.
weighting	the weights to be used when normalizing scores (see 'Details').
rowsup	a matrix with the same columns as the model data giving supplementary (passive) rows to include in the result.
colsup	a matrix with the same rows as the model data giving supplementary (passive) columns to include in the result.
...	currently unused.

Details

These functions extract parameters from gnm log-multiplicative models and make them identifiable by imposing the required constraints on them. The general pattern is that row and column scores are separately centered around 0 and scaled so that they sum to 1, and so that their cross-dimensional correlation is null. From this operation result two series of scores (rows and columns) plus an intrinsic association coefficient (phi) for each dimension.

Most users do not need to call these directly, but they are still made public since they may be useful for advanced uses, notably when combining log-multiplicative association components with other model specifications. assoc can be used to identify the scores, the rest of the coefficients being extracted manually by the caller.

Value

An assoc object with the following components:

phi	The intrisic association parameters, one per dimension.
row	Row scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
col	Column scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.

`weighting`	The name of the weighting method used, reflected by `row.weights` and `col.weights`.
`row.weights`	The row weights used for the identification of scores, as specified by the `weighting` argument.
`col.weights`	The column weights used for the identification of scores, as specified by the `weighting` argument.

See Also

`rc`, `hmskew`, `yrcskew`, `rcL`, `rcL.trans`

`color` *Two Cross-Classifications of Eye Color by Hair Color*

Description

Three-way table crossing eye color and hair color in two places, Caithness and Aberdeen. This table is used by Becker and Clogg (1989) to illustrate several log-multiplicative row-column association models, with and without layer effect.

Usage

```
data(color)
```

References

Becker, M.P., and Clogg, C.C. (1989). Analysis of Sets of Two-Way Contingency Tables Using Association Models. *J. of the Am. Stat. Association* 84(405), 142-151.

Examples

```
## see ?rc
```

`criminal` *Dropped Criminal Charges, Denmark, 1955-1958*

Description

Number of men aged 15-19 charged with a criminal case for whom charges were dropped: Denmark, 1955-1958. This two-way table is used by Goodman (1991) to illustrate a log-multiplicative row-column model with one dimension. It was used before by Rasch (1966), Christiansen and Stene (1969), and Andersen (1986, 1990).

Usage

```
data(criminal)
```

References

Goodman, L.A. (1991). Measures, Models, and Graphical Displays in the Analysis of Cross-Classified Data. *J. of the Am. Stat. Association* 86(416), 1086, Table 1.

Examples

```
## see ?rc
```

`gss7590` *Education and Occupational Attainment Among White Men and Women in the United States, 1975-1990*

Description

Three-way table crossing education and occupational attainment by sex and period among white men and women from the General Social Survey: United States, 1975-1980 and 1985-1990. This table is used by Wong (2010) to illustrate log-multiplicative row-column models with three dimensions.

Usage

```
data(gss7590)
```

References

Wong, R.S-K. (2010). Association models. SAGE. 32, Table 4.3.

Examples

```
## see ?rcL and ?plot.rcL
```

`gss8590` *Education and Occupational Attainment Among Women in the United States, 1985-1990*

Description

Two-way table crossing education and occupational attainment among women from the General Social Survey: United States, 1985-1990. This table is used by Wong (2001, 2010) to illustrate a log-multiplicative row-column model with two dimensions.

Usage

```
data(gss8590)
```

References

Wong, R.S-K. (2001). Multidimensional Association Models : A Multilinear Approach. *Sociol. Methods & Research* 30, 197-240, Table 2.

Wong, R.S-K. (2010). *Association models*. SAGE. 32, Table 2.3 B.

Examples

```
## see ?rc and ?plot.rc
# The table reported in Wong (2010) has a cell inconsistent with
# what was reported in Wong (2001). To fix this:
data(gss8590)
tab <- margin.table(gss8590[,,c(2,4)], 1:2)
tab[2,4] <- 49
```

---

gss88 *Major Occupation by Years of Schooling in the United States, 1988*

Description

Two-way table crossing occupational attainment and years of education among persons aged 20 through 64 from the General Social Survey: United States, 1988. This table is used by Clogg and Shihadeh (1994) to illustrate a log-multiplicative row-column model with one dimension.

Usage

```
data(gss88)
```

References

Clogg, C.C., and Shihadeh, E.S. (1994). Statistical Models for Ordinal Variables. Sage: Advanced Quantitative Techniques in the Social Sciences (4), Table 3.1, p. 40.

Examples

```
## see ?rc
```

hg16 — *Son's occupation by father's occupation for 16 countries in the 1960s and 1970s*

Description

Three-way mobility table assembled by Hazelrigg and Garnier (1976), and used by Zhou (2015) to illustrate the shrinkage estimation of the log-odds ratios and of the Altham index.

Usage

```
data(hg16)
```

References

Hazelrigg, L.E., Garnier, M.A (1976). Occupational Mobility in Industrial Societies: A Comparative Analysis of Differential Access to Occupational Ranks in Seventeen Countries. *American Sociological Review* 41: 498-511.

Zhou, X. (2015). Shrinkage Estimation of Log-Odds Ratios for Comparing Mobility Tables. *Sociological Methodology* 45(1):33-63.

Examples

```
## see ?iac
```

HMSkew — *Specify a Skew-Symmetric Association in a gnm Model Formula*

Description

A function of class `"nonlin"` to specify a van der Heijden & Mooijaart (1995) skew-symmetric association in the formula argument to `gnm`.

Usage

```
HMSkew(row, col, inst = NULL)
```

Arguments

`row`	the levels of the row variable
`col`	the levels of the column variable
`inst`	a positive integer specifying the instance number of the term

Details

This function is used by hmskew to fit skew-symmetric models proposed by van der Heijden & Mooijaart (1995) and their variants. It can be used directly to fit custom variants of the model not supported by hmskew.

This function combines its arguments in the following way:

$$HMSkew(i,j) = \nu_i\mu_j - \mu_i\nu_j$$

where $HMSkew(i,j)$ is the skew association for the cell at the intersection of row i and column j of the table. See reference for mathematical details.

Value

A list with the required components of a "nonlin" function:

predictors	the expressions passed to Mult
term	a function to create a deparsed mathematical expression of the term, given labels for the predictors.
call	the call to use as a prefix for parameter labels.

Author(s)

Milan Bouchet-Valat

References

van der Heijden, P.G.M., and Mooijaart, A. (1995). Some new log bilinear models for the analysis of asymmetry in a square contingency table. *Sociol. Methods and Research* 24, 7-29.

See Also

hmskew

Examples

```
# See ?hmskew.
```

---

hmskew *Fitting van der Heijden & Mooijaart Skew-Symmetric Association Model*

Description

Fits a skew-symmetric association model proposed in van der Heijden & Mooijaart (1995) to describe asymmetry of square tables. Skew-symmetric association can be combined with quasi-symmetry (the default), quasi-independence, or symmetric (homogeneous) RC(M) associations.

Usage

```
hmskew(tab, nd.symm = NA, diagonal = FALSE,
       weighting = c("marginal", "uniform", "none"),
       rowsup = NULL, colsup = NULL,
       se = c("none", "jackknife", "bootstrap"),
       nreplicates = 100, ncpus = getOption("boot.ncpus"),
       family = poisson, weights = NULL,
       start = NULL, etastart = NULL, tolerance = 1e-8,
       iterMax = 5000, trace = FALSE, verbose = TRUE, ...)
```

Arguments

tab	a square two-way table, or an object (such as a matrix) that can be coerced into a table; if present, dimensions above two will be collapsed.
nd.symm	the number of dimensions to include in the *symmetric* RC(M) association. Cannot exceed `2 * min(nrow(tab) - 1, ncol(tab) - 1)` (quasi-symmetry model). If NA (the default), a full quasi-symmetric association is used instead of a RC(M) model; if 0, quasi-independence is used.
diagonal	should the model include parameters specific to each diagonal cell? This amounts to taking quasi-independence, rather than independence, as the baseline model.
weighting	what weights should be used when normalizing the scores.
rowsup	if present, a matrix with the same columns as tab and rows corresponding to the columns of colsup, giving supplementary (passive) rows.
colsup	if present, a matrix with the same rows as tab and columns corresponding to the rows of colsup, giving supplementary (passive) columns.
se	which method to use to compute standard errors for parameters.
nreplicates	the number of bootstrap replicates, if enabled.
ncpus	the number of processes to use for jackknife or bootstrap parallel computing. Defaults to the number of cores (see detectCores), with a maximum of 5, but falls back to 1 (no parallelization) if package parallel is not available.
family	a specification of the error distribution and link function to be used in the model. This can be a character string naming a family function; a family function, or the result of a call to a family function. See family details of family functions.
weights	an optional vector of weights to be used in the fitting process.
start	either NA to use optimal starting values, NULL to use random starting values, or a vector of starting values for the parameters in the model.
etastart	starting values for the linear predictor; set to NULL to use either default starting values (if start = NA), or random starting values (in all other cases).
tolerance	a positive numeric value specifying the tolerance level for convergence; higher values will speed up the fitting process, but beware of numerical instability of estimated scores!
iterMax	a positive integer specifying the maximum number of main iterations to perform; consider raising this value if your model does not converge.

trace	a logical value indicating whether the deviance should be printed after each iteration.
verbose	a logical value indicating whether progress indicators should be printed, including a diagnostic error message if the algorithm restarts.
...	more arguments to be passed to gnm

Details

The original model presented by van der Heijden & Mooijaart (1995), called "quasi-symmetry plus skew-symmetry", combines a skew-symmetric association with a quasi-symmetry baseline; it is the variant fitted by default by this function. If nd.symm is set to a positive integer value, though, variants using a RC(M) model to describe the *symmetric association* are used, with our without diagonal-specific parameters (depending on the value of the diagonal argument).

These models follow the equation:

$$logF_{ij} = q_{ij} + \phi(\nu_i\mu_j - \mu_i\nu_j)$$

where F_{ij} is the expected frequency for the cell at the intersection of row i and column j of tab, and q_{ij} a quasi-symmetric specification, with either full interaction parameters, or a RC(M) association. See reference for detailed information about the degrees of freedom and the identification constraints applied to the scores.

Another model presented in the paper, the "symmetry plus skew-symmetry model" is not currently supported out of the box, but should be relatively straightforward to implement using the underlying assoc.hmskew function combined with a symmetric association model.

Actual model fitting is performed using gnm, which implements the Newton-Raphson algorithm. This function simply ensures correct start values are used, in addition to allowing for identification of scores even with several dimensions, computation of their jackknife or bootstrap standard errors, and plotting. The default starting values for skew association parameters are computed using an eigen value decomposition from the results of the model without skew association component ("base model"); if nd.symm is not NA and strictly positive, random starting values are used. In some complex cases, using start = NULL to start with random values can be more efficient, but it is also less stable and can converge to non-optimal solutions.

Value

A hmskew object, which is a subclass of an rc.symm object (see rc) if nd.symm is strictly positive. In addition to this class, it contains a assoc.hmskew component holding information about the *skew-symmetric* association:

phi	The intrisic association parameters, one per dimension.
row	Row scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
col	Column scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
row.weights	The row weights used for the identification of scores, as specified by the weighting argument.

col.weights	The column weights used for the identification of scores, as specified by the weighting argument.
covmat	The variance-covariance matrix for phi coefficients and normalized row and column scores. Only present if se was not "none".
adj.covmats	An array stacking on its third dimension one variance-covariance matrix for the adjusted scores of each layer in the model (used for plotting). Only present if se was not "none".
covtype	The method used to compute the variance-covariance matrix (corresponding to the se argument.

Author(s)

Milan Bouchet-Valat

References

van der Heijden, P.G.M., and Mooijaart, A. (1995). Some new log bilinear models for the analysis of asymmetry in a square contingency table. *Sociol. Methods and Research* 24, 7-29.

See Also

plot.hmskew, gnm

Examples

```
## van der Heijden & Mooijaart (1995), Table 2c, p. 23
data(ocg1973)

# 5:1 is here to take "Farmers" as reference category (angle 0)
model <- hmskew(ocg1973[5:1, 5:1], weighting="uniform")
model
ass <- model$assoc.hmskew

# First column of the table
round(ass$row[,,1] * sqrt(ass$phi[1,1]), d=2)[5:1,]

# Right part of the table
round(ass$phi[1] * (ass$row[,2,1] %o% ass$row[,1,1] -
                    ass$row[,1,1] %o% ass$row[,2,1]), d=3)[5:1, 5:1]

# Plot
plot(model, coords="cartesian")
```

`hmskewL` *Fitting van der Heijen & Mooijaart Skew-Symmetric Association Model With Layer Effect*

Description

Fits an extension of the skew-symmetric association model proposed in van der Heijden & Mooijaart (1995) to describe asymmetry of square tables. This model introduces a layer effect by which the strength of skew-symmetric association, and optionnally scores, can vary over the levels of the third dimension of the table. Skew-symmetric association can be combined with quasi-symmetry (the default), quasi-independence, or symmetric (homogeneous) RC(M) associations, with or without layer effect.

Usage

```
hmskewL(tab, nd.symm = NA,
        layer.effect.skew = c("homogeneous.scores", "heterogeneous",
                              "none"),
        layer.effect.symm = c("heterogeneous", "uniform",
                              "regression.type",
                              "homogeneous.scores", "none"),
        diagonal = c("none", "heterogeneous", "homogeneous"),
        weighting = c("marginal", "uniform", "none"),
        se = c("none", "jackknife", "bootstrap"),
        nreplicates = 100, ncpus = getOption("boot.ncpus"),
        family = poisson, weights = NULL,
        start = NULL, etastart = NULL, tolerance = 1e-8,
        iterMax = 5000, trace = FALSE, verbose = TRUE, ...)
```

Arguments

`tab`	a three-way table, or an object (such as a matrix) that can be coerced into a table; if present, dimensions above three will be collapsed. First two dimensions must be symmetric (i.e. of the same length).
`nd.symm`	the number of dimensions to include in the *symmetric* RC(M) association. Cannot exceed `2 * min(nrow(tab) - 1, ncol(tab) - 1)` (quasi-symmetry model). If `NA` (the default), a full quasi-symmetric association is used instead of a RC(M) model; if `0`, quasi-independence is used.
`layer.effect.skew`	determines the form of the interaction between skew-symmetric association and layers. See "Details" below.
`layer.effect.symm`	determines the form of the interaction between symmetric row-column association, or quasi-symmetric association (if `nd.symm = NA`) and layers. See "Details" below.

diagonal	what type of diagonal-specific parameters to include in the model, if any. Only makes sense when nd.symm is not NA (else, diagonal parameters are already included).
weighting	what weights should be used when normalizing the scores.
se	which method to use to compute standard errors for parameters.
nreplicates	the number of bootstrap replicates, if enabled.
ncpus	the number of processes to use for jackknife or bootstrap parallel computing. Defaults to the number of cores (see detectCores), with a maximum of 5, but falls back to 1 (no parallelization) if package parallel is not available.
family	a specification of the error distribution and link function to be used in the model. This can be a character string naming a family function; a family function, or the result of a call to a family function. See family details of family functions.
weights	an optional vector of weights to be used in the fitting process.
start	either NA to use optimal starting values, NULL to use random starting values, or a vector of starting values for the parameters in the model.
etastart	starting values for the linear predictor; set to NULL to use either default starting values (if start = NA), or random starting values (in all other cases).
tolerance	a positive numeric value specifying the tolerance level for convergence; higher values will speed up the fitting process, but beware of numerical instability of estimated scores!
iterMax	a positive integer specifying the maximum number of main iterations to perform; consider raising this value if your model does not converge.
trace	a logical value indicating whether the deviance should be printed after each iteration.
verbose	a logical value indicating whether progress indicators should be printed, including a diagnostic error message if the algorithm restarts.
...	more arguments to be passed to gnm

Details

This model follows an equation inspired from that presented by van der Heijden & Mooijaart (1995) for two-way tables (see hmskew):

$$logF_{ijk} = q_{ijk} + \phi_k(\nu_{ik}\mu_{jk} - \mu_{ik}\nu_{jk})$$

where F_{ijk} is the expected frequency for the cell at the intersection of row i, column j and layer k of tab, and q_{ij} a quasi-symmetric specification, with either full interaction parameters, or a RC(M) association. See reference for detailed information about the degrees of freedom and the identification constraints applied to the scores.

If layer.effect.skew is set to 'heterogeneous', different scores will be computed for each level, which is equivalent to fitting separate models using hmskew on the k two-way tables. If it is set to 'homogeneous.scores', then $\mu_{ik} = \mu_i$ and $\nu_{ik} = \nu_i$ for all layers k: only the ϕ_k are allowed to vary across layers. If it is set to 'none', then in addition to the previous conditions all ϕ_{mk} are forced to be equal for all layers k, which amounts to a stability of the association across layers.

When nd.symm is different from NA, the symmetric association works exactly like a call to rcL, with parameters nd.symm and layer.effect.symm translated respectively to nd and layer.effect. When nd.symm == NA, symmetric association parameters are either stable across layers, are multiplied by a layer coefficient (UNIDIFF model, see unidiff), follow a regression-type (Goodman-Hout) specification, or are different for each layer, when layer.effect.symm is respectively none, uniform, regression.type and heterogeneous.

Actual model fitting is performed using gnm, which implements the Newton-Raphson algorithm. This function simply ensures correct start values are used, in addition to allowing for identification of scores even with several dimensions, computation of their jackknife or bootstrap standard errors, and plotting. The default starting values for skew association parameters are computed using an eigen value decomposition from the results of the model without skew association component ("base model"); if nd.symm is not NA and strictly positive, random starting values are used. In some complex cases, using start = NULL to start with random values can be more efficient, but it is also less stable and can converge to non-optimal solutions.

Value

A hmskewL object, which is a subclass of an rcL.symm object (see rcL) if nd.symm is strictly positive. In addition to this class, it contains a assoc.hmskew component holding information about the *skew-symmetric* association:

phi	The intrisic association parameters, one per dimension and per layer.
row	Row scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
col	Column scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
weighting	The name of the weighting method used, reflected by row.weights and col.weights.
row.weights	The row weights used for the identification of scores, as specified by the weighting argument.
col.weights	The column weights used for the identification of scores, as specified by the weighting argument.
covmat	The variance-covariance matrix for phi coefficients and normalized row and column scores. Only present if se was not "none".
adj.covmats	An array stacking on its third dimension one variance-covariance matrix for the adjusted scores of each layer in the model (used for plotting). Only present if se was not "none".
covtype	The method used to compute the variance-covariance matrix (corresponding to the se argument.

Author(s)

Milan Bouchet-Valat

References

van der Heijden, P.G.M., and Mooijaart, A. (1995). Some new log bilinear models for the analysis of asymmetry in a square contingency table. *Sociol. Methods and Research* 24, 7-29.

See Also

plot.hmskewL, hmskew, gnm

---

iac *Intrinsic Association Coefficient*

Description

Compute the intrisic association coefficient of a table. This coefficient was first devised by Goodman (1996) as the "generalized contingency" when a logarithm link is used, and it is equal to the standard deviation of the log-linear two-way interaction parameters λ_{ij}. To obtain the Altham index, multiply the result by `sqrt(nrow(tab) * ncol(tab)) * 2` (see "Examples" below).

Usage

```
iac(tab, cell = FALSE,
    weighting = c("marginal", "uniform", "none"),
    component = c("total", "symmetric", "antisymmetric"),
    shrink = FALSE,
    normalize = FALSE,
    row.weights = NULL, col.weights = NULL)
```

Arguments

tab	a two- or three-way table without zero cells; for three-way tables, average marginal weighting is used when "weighting = "marginal"", and the MAOR is computed for each layer (third dimension).
cell	if "TRUE", return the per-cell contributions (affected by the value of phi, see "Details" below).
weighting	what weights should be used when normalizing the scores.
component	whether to compute the total association, or from symmetric or antisymmetric interaction coefficients only.
shrink	whether to use the empirical Bayes shrinkage estimator proposed by Zhou (2015) rather than the direct estimator.
normalize	whether to return the normalized version of the index varying between 0 and 1 proposed by Bouchet-Valat (2022) rather than the classic index varying between 0 and positive infinity.
row.weights	optional custom weights to be used for rows, e.g. to compute the phi coefficient for several tables using their overall marginal distribution. If specified, weighting is ignored.
col.weights	see row.weights.

Details

See Goodman (1996), Equation 52 for the (marginal or other) weighted version of the intrinsic association coefficient ($\tilde{\lambda}$); the unweighted version can be computed with unit weights. The coefficient should not be confused with Goodman and Kruskal's lambda coefficient. The uniform-weighted version is defined as:

$$\lambda^{\dagger} = \sqrt{\frac{1}{IJ}\sum_{i=1}^{I}\sum_{j=1}^{J}\lambda_{ij}^2}$$

The (marginal or other) weighted version is defined as:

$$\tilde{\lambda} = \sqrt{\sum_{i=1}^{I}\sum_{j=1}^{J}\tilde{\lambda}_{ij}^2 P_{i+}P_{+j}}$$

with $\sum_{i=1}^{I}\lambda_{ij} = \sum_{j=1}^{J}\lambda_{ij} = 0$ and $\sum_{i=1}^{I}P_{i+}\tilde{\lambda}_{ij} = \sum_{j=1}^{J}P_{+j}\tilde{\lambda}_{ij} = 0$.

The normalized version of the index is defined from $\lambda^{\dagger}$ and $\tilde{\lambda}$ as:

$$\tau = \sqrt{1 + 1/(2\lambda)^2} - 1/(2\lambda)$$

Per-cell contributions c_{ij} are defined so that: $\tilde{\phi} = \sqrt{\sum_{i=1}^{I}\sum_{j=1}^{J}c_{ij}}$. For the unweighted case, $c_{ij} = \lambda_{ij}^2/IJ$; for the weighted case, $\tilde{c}_{ij} = \tilde{\lambda}_{ij}^2 P_{i+}P_{+j}$.

This index cannot be computed in the presence of zero cells since it is based on the logarithm of proportions. In these cases, 0.5 is added to all cells of the table (Agresti 2002, sec. 9.8.7, p. 397; Berkson 1955), and a warning is printed. Make sure this correction does not affect too much the results (especially with small samples) by manually adding different values before calling this function.

Value

The numeric value of the intrinsic association coefficient (if `cell = FALSE`), or the corresponding per-cell contributions (if `cell = TRUE`).

Author(s)

Milan Bouchet-Valat

References

Agresti, A. 2002. Categorical Data Analysis. New York: Wiley.

Altham, P. M. E., Ferrie J. P., 2007. Comparing Contingency Tables Tools for Analyzing Data from Two Groups Cross-Classified by Two Characteristics. *Historical Methods* 40(1):3-16.

Bouchet-Valat, M. (2022). General Marginal-free Association Indices for Contingency Tables: From the Altham Index to the Intrinsic Association Coefficient. *Sociological Methods & Research* 51(1): 203-236.

Berkson, J. (1955). Maximum Likelihood and Minimum chi2 Estimates of the Logistic Function. *J. of the Am. Stat. Ass.* 50(269):130-162.

Goodman, L. A. (1996). A Single General Method for the Analysis of Cross-Classified Data: Reconciliation and Synthesis of Some Methods of Pearson, Yule, and Fisher, and Also Some Methods of Correspondence Analysis and Association Analysis. *J. of the Am. Stat. Ass.* 91(433):408-428.

Zhou, X. (2015). Shrinkage Estimation of Log-Odds Ratios for Comparing Mobility Tables. *Sociological Methodology* 45(1):33-63.

See Also

unidiff, rc, maor

Examples

```
# Altham index (Altham and Ferrie, 2007, Table 1, p. 3 and commentary p. 8)
tab1 <- matrix(c(260, 195, 158, 70,
                 715, 3245, 874, 664,
                 424, 454, 751, 246,
                 142, 247, 327, 228), 4, 4)
iac(tab1, weighting="n") * sqrt(nrow(tab1) * ncol(tab1)) * 2

# Zhou (2015)
data(hg16)
# Add 0.5 due to the presence of zero cells
hg16 <- hg16 + 0.5
# Figure 3, p. 343: left column then right column
# (reported values are actually twice the Altham index)
iac(hg16, weighting="n") * sqrt(nrow(hg16) * ncol(hg16)) * 2 * 2
iac(hg16, weighting="n", shrink=TRUE) * sqrt(nrow(hg16) * ncol(hg16)) * 2 * 2
# Table 4, p. 347: values are not exactly the same
u <- unidiff(hg16)
# First row
cor(u$unidiff$layer$qvframe$estimate, iac(hg16, weighting="n"))
cor(u$unidiff$layer$qvframe$estimate, iac(hg16, weighting="n"), method="spearman")
# Second row
cor(u$unidiff$layer$qvframe$estimate, iac(hg16, shrink=TRUE, weighting="n"))
cor(u$unidiff$layer$qvframe$estimate, iac(hg16, shrink=TRUE, weighting="n"), method="spearman")
```

---

maor *Mean Absolute Odds Ratio or Intrinsic Association Coefficient*

Description

Compute the mean absolute odds ratio of a table, i.e. the (possibly weighted) geometric mean of the odds ratios or of their inverse when they are above one, which is also closely related to the the intrinsic association coefficient. The latter coefficient was first devised by Goodman (1996) as the "generalized contingency" when a logarithm link is used, and it is equal to the mean of the absolute value of log-linear two-way interaction parameters λ_{ij} (in its original version it consists in the square root of the sum of squared parameters).

Usage

```
maor(tab, phi = FALSE, cell = FALSE,
     weighting = c("marginal", "uniform", "none"),
     norm = 2, component=c("total", "symmetric", "antisymmetric"),
     row.weights = NULL, col.weights = NULL)
```

Arguments

tab	a two- or three-way table without zero cells; for three-way tables, average marginal weighting is used when "weighting = "marginal"", and the MAOR is computed for each layer (third dimension).
phi	if "TRUE", return the intrinsic association coefficient rather than the Mean absolute odds ratio.
cell	if "TRUE", return the per-cell contributions (affected by the value of phi, see "Details" below).
weighting	what weights should be used when normalizing the scores.
norm	the norm to use to compute the mean of λ_{ij} parameters, 1 for the mean of absolute values, or 2 for the square root of the sum of squared parameters (as in the original version).
component	whether to compute the total association, or from symmetric or antisymmetric interaction coefficients only.
row.weights	optional custom weights to be used for rows, e.g. to compute the phi coefficient for several tables using their overall marginal distribution. If specified, weighting is ignored.
col.weights	see row.weights.

Details

See Goodman (1996), Equation 52 for the (marginal or other) weighted version of the intrinsic association coefficient ($\tilde{\phi}$); the unweighted version can be computed with unit weights. The coefficient is called $\tilde{\lambda}^2$ in the original article, but to avoid the confusion with Goodman and Kruskal's lambda coefficient, it is here denoted as ϕ, as usual in row-column association models. The uniform-weighted version is defined as:

$$\phi = \sqrt{\frac{1}{IJ}\sum_{i=1}^{I}\sum_{j=1}^{J}\lambda_{ij}^2}$$

The (marginal or other) weighted version is defined as:

$$\tilde{\phi} = \sqrt{\sum_{i=1}^{I}\sum_{j=1}^{J}\tilde{\lambda}_{ij}^2 P_{i+}P_{+j}}$$

with $\sum_{i=1}^{I}\lambda_{ij} = \sum_{j=1}^{J}\lambda_{ij} = 0$ and $\sum_{i=1}^{I}P_{i+}\tilde{\lambda}_{ij} = \sum_{j=1}^{J}P_{+j}\tilde{\lambda}_{ij} = 0$.

The uniform-weighted version of the mean absolute odds ratio (MAOR) is defined as:

$$MAOR = \exp\left[\sqrt{\frac{4}{IJ(I-1)(J-1)}}\phi\right]$$

The (marginal or other) weighted version is defined as:

$$MAOR = \exp\left[\sqrt{\frac{4}{\sum_{i=1}^{I}\sum_{j=1}^{J} P_{i+}(1-P_{i+})P_{+j}(1-P_{+j})}}\tilde{\phi}\right]$$

Per-cell contributions c_{ij} are defined so that $\tilde{\phi} = \sqrt{\sum_{i=1}^{I}\sum_{j=1}^{J} c_{ij}}$
and $MAOR = exp\left[\sqrt{\sum_{i=1}^{I}\sum_{j=1}^{J} c_{ij}}\right]$.

This index cannot be computed in the presence of zero cells since it is based on the logarithm of proportions. In these cases, 0.5 is added to all cells of the table (Agresti 2002, sec. 9.8.7, p. 397; Berkson 1955), and a warning is printed. Make sure this correction does not affect too much the results (especially with small samples) by manually adding different values before calling this function.

Value

The numeric value of the mean absolute odds ratio, or of the intrinsic association coefficient (if `phi = TRUE`), or the corresponding per-cell contributions (if `cell = TRUE`).

Author(s)

Milan Bouchet-Valat

References

Agresti, A. 2002. Categorical Data Analysis. New York: Wiley.

Goodman, L. A. (1996). A Single General Method for the Analysis of Cross-Classified Data: Reconciliation and Synthesis of Some Methods of Pearson, Yule, and Fisher, and Also Some Methods of Correspondence Analysis and Association Analysis. *J. of the Am. Stat. Ass.* 91(433):408-428.

Berkson, J. (1955). Maximum Likelihood and Minimum chi2 Estimates of the Logistic Function. *J. of the Am. Stat. Ass.* 50(269):130-162.

See Also

`iac`, `unidiff`, `rc`

---

ocg1973 *Intergenerational Mobility in the United States, 1973*

Description

Mobility table for the United States from the 1973 Occupational Changes in a Generation (OCG-II) survey. This table has been used by Yamaguchi (1987, 1990), Xie (1992) and van der Heijden & Mooijaart (1995).

Usage

```
data(ocg1973)
```

References

Yamaguchi, K. (1990). Some Models for the Analysis of Asymmetric Association in Square Contingency Tables with Ordered Categories. *Sociological Methodology* 20, 181-212.

van der Heijden, P.G.M., and Mooijaart, A. (1995). Some new log bilinear models for the analysis of asymmetry in a square contingency table. *Sociol. Methods and Research* 24, 7-29.

Examples

```
## see ?yrcskew, ?hmskew and ?plot.hmskew
```

---

plot.assoc *Plotting Scores from Association Models*

Description

Graphical display of category scores from association models.

Usage

```
## S3 method for class 'rc'
plot(x, dim = c(1, 2),
     what = c("both", "rows", "columns"), which = NULL,
     mass = TRUE, luminosity = length(x$assoc$diagonal > 0),
     conf.int = NA, replicates = FALSE,
     coords = c("cartesian", "polar"), rev.axes = c(FALSE, FALSE),
     cex = par("cex"), col = c("blue", "red"), col.conf.int = col, groups = NULL,
     add = FALSE, type, xlim, ylim, asp, xlab, ylab, main, pch, font, ...)

## S3 method for class 'rc.symm'
plot(x, dim = c(1, 2), which = NULL,
     mass = TRUE, luminosity = length(x$assoc$diagonal > 0),
     conf.int = NA, replicates = FALSE,
     coords = c("cartesian", "polar"), rev.axes = c(FALSE, FALSE),
     cex = par("cex"), col = "blue", col.conf.int = col, groups = NULL,
     add = FALSE, type, xlim, ylim, asp, xlab, ylab, main, pch, font, ...)

## S3 method for class 'hmskew'
plot(x, dim = c(1, 2),
     what = c("skew-symmetric", "symmetric"), which = NULL,
     mass = TRUE, luminosity = length(x$assoc.hmskew$diagonal > 0),
     arrow = 45, conf.int = NA, replicates = FALSE,
     coords = c("polar", "cartesian"), rev.axes = c(FALSE, FALSE),
     cex = par("cex"), col = "blue", col.conf.int = col, groups = NULL,
```

```
     add = FALSE, type, xlim, ylim, asp, xlab, ylab, main, pch, font, ...)

## S3 method for class 'yrcskew'
plot(x, dim = c(1, 2),
     what = c("skew-symmetric", "symmetric"), which = NULL,
     mass = TRUE, luminosity = length(x$assoc.yrcskew$diagonal > 0),
     arrow = 45, conf.int = NA, replicates = FALSE,
     coords = c("polar", "cartesian"), rev.axes = c(FALSE, FALSE),
     cex = par("cex"), col = "blue", col.conf.int = col, groups = NULL,
     add = FALSE, type, xlim, ylim, asp, xlab, ylab, main, pch, font, ...)

## S3 method for class 'rcL'
plot(x, dim = c(1, 2), layer = "average",
     what = c("both", "rows", "columns"), which = NULL,
     mass = TRUE, luminosity = length(x$assoc$diagonal > 0),
     conf.int = NA, replicates = FALSE,
     coords = c("cartesian", "polar"), rev.axes = c(FALSE, FALSE),
    cex = par("cex"), col = c("blue", "red"), col.conf.int = col, groups = NULL,
     add = FALSE, type, xlim, ylim, asp, xlab, ylab, main, pch, font, ...)

## S3 method for class 'rcL.symm'
plot(x, dim = c(1, 2), layer = "average",
     which = NULL,
     mass = TRUE, luminosity = length(x$assoc$diagonal > 0),
     conf.int = NA, replicates = FALSE,
     coords = c("cartesian", "polar"), rev.axes = c(FALSE, FALSE),
     cex = par("cex"), col = "blue", col.conf.int = col, groups = NULL,
     add = FALSE, type, xlim, ylim, asp, xlab, ylab, main, pch, font, ...)

## S3 method for class 'hmskewL'
plot(x, dim = c(1, 2), layer = "average",
     what = c("skew-symmetric", "symmetric"), which = NULL,
     mass = TRUE, luminosity = length(x$assoc.hmskew$diagonal > 0),
     arrow=45, conf.int = NA, replicates = FALSE,
     coords = c("polar", "cartesian"), rev.axes = c(FALSE, FALSE),
     cex = par("cex"), col = "blue", col.conf.int = col, groups = NULL,
     add = FALSE, type, xlim, ylim, asp, xlab, ylab, main, pch, font, ...)

## S3 method for class 'assoc'
plot(x, dim = c(1, 2), layer = 1,
     what = c("both", "rows", "columns"), which = NULL,
     mass = TRUE, luminosity = length(x$diagonal > 0),
     arrow = NULL, conf.int = NA, replicates = FALSE,
     coords = c("cartesian", "polar"), rev.axes = c(FALSE, FALSE),
    cex = par("cex"), col = c("blue", "red"), col.conf.int = col, groups = NULL,
     add = FALSE, type, xlim, ylim, asp, xlab, ylab, main, pch, font, ...)
```

Arguments

x	an association model, or an object inheriting from class assoc.
dim	numerical vector of length 2 indicating the dimensions to plot on horizontal and vertical axes respectively; default is first dimension horizontal and second dimension vertical.
layer	integer indicating which layer should be represented, or "average" or "average.rotate" when scores are homogeneous (see "Details" below).
what	for rc and assoc objects, whether points corresponding to rows, columns or both should be plotted; for hmskew and yrcskew objects, what association should be plotted.
which	an optional subset of points to be plotted, specified via a logical, integer or character vector indexing the row or column component of the association object; if what = both, a list of two such vectors, resp. for rows and columns.
mass	whether the size of the point symbols should reflect on the mass of the categories; this only makes sense when marginal weights were used when fitting the model. The precise formula is that the pch of a symbol is equal to the pch argument, times the weight of the category divided by average weight.
luminosity	whether the luminosity of the symbols should vary according to the diagonal-specific parameters (if diagonal = TRUE was passed when fitting the model); if TRUE, hue and saturation of col are taken as a base, and value varies from 0 to 0.8 as a linear function of the diagonal parameter values.
arrow	Numeric value indicating the angle at which the polar coordinates system arrow should be plotted; use NULL for no arrow.
conf.int	an integer between 0 and 1 giving the confidence level to use when drawing error bars/ellipses around the points (see "Details" below); by default nothing is plotted. Only possible if jackknife or bootstrap was enabled when fitting the model.
replicates	whether to plot points representing the values of the scores obtained for all of the jackknife of bootrap replicates, when applicable (see "Details" below).
coords	whether to use a Cartesian or a polar coordinate system; the former makes sense when axes offer an interpretation (like in RC(M) models), while the latter are more appropriate when only the angle and distance to origin are of interest (like in hmskew models).
rev.axes	a numeric of length 1 or 2 indicating whether the sign of scores on the axes should be changed; as this sign is arbitrary in RC(M) models, changing it has no incidence on the results and may be more intuitive or consistent with other presentations.
cex	a numeric vector indicating the size of the point symbols, recycled as necessary; the most common choice is probably to pass only one value and use mass to make the size vary.
col	a vector indicating the color of the point symbols, recycled as necessary; as a special case, a vector of length 2 can be passed, to indicate the color of row and column points, respectively. See also luminosity.

col.conf.int	a vector indicating the color of the confidence bars/ellipses, when these are plotted; see col the format.
groups	a vector indicating what symbol should be used for each point, recycled as necessary; groups will use pch values 21, 24, 22, 23 and 25, in this order, cycling if needed. If not an integer, the number of the factor level will be used.
add	whether to draw over an existing plot instead of creating a new one.
type	set to "n" to avoid actually plotting the points and labels; useful for customization based on the returned coordinates, see "Value" below.
xlim	numeric vector of length 2, giving the x coordinates range.
ylim	numeric vector of length 2, giving the y coordinates range.
asp	the y/x aspect ratio, see plot.window.
xlab	a title for the x axis: see title. For RC(M) axes and Cartesian coordinates, the default is "Dimension N (phi)"; it is empty in other cases where axes have no meaning.
ylab	a title for the y axis: see title. For RC(M) axes and Cartesian coordinates, the default is "Dimension N (phi)"; it is empty in other cases where axes have no meaning.
main	an overall title for the plot: see title. If missing for RC(M)-L models, the name of the plotted layer will be used.
pch	a vector of plotting 'character', i.e., symbol to use for each point, recycled as necessary; see points.
font	an integer vector indicating the font to use for each label, recycled as necessary; see link{par}.
...	Further arguments passed to plot.

Details

The functions documented here represent in a one- or two-dimensional space the category scores obtained from a log-multiplicative association model. They produce *symmetric biplots* in which the coordinates of points on both axes are the product of normalized scores and of the square root of the intrinsic association coefficient corresponding to each dimension: thus, row and column points share the same "unit" on all axes (Goodman, 1991, Appendix 2; Wong, 2010, eq. 2.38; Clogg & Shihadeh, 1994, p. 91-92). As a special case, models with only one dimension are presented as a dotchart of the scores.

Various convenience options are provided, with reasonable defaults for each model family. In particular, you may find it necessary to adapt the cex, mass, luminosity and groups arguments depending on the number of categories to be plotted and to their respective weights. When plotting a RC(2) model, a polar coordinate system can be of substantive interest, allowing to interpret at a glance the distance to origin as the general strength of the association for a category on both axes (a property that is lost for higher-dimensional models).

Confidence bars/ellipses are computed from the scores' variances and covariances, based on the assumption that they are follow a normal distribution, even if standard errors are computed using jackknife or bootstrap. When bootstrap (not jackknife) was used, this normality assumption can be assessed visually using the replicates argument to check whether points globally follow the

shape of the ellipses. See se.assoc for details about checking the validity of jackknife or bootstrap results.

When layer is set to "average" for models with layer effect and homogeneous scores, intrinsic association coefficients are weighted across all layers. In addition, if layer is set to "average.rotate", scores are rotated so that axes of the plot are those with the highest variance; oblique axes represent the original dimensions in the new space.

The plot.assoc function is called internally by all others, and may be leveraged for advanced use cases, like plotting custom models that do not correspond stricly to the supported types.

Value

An invisible list with components row and col, two matrices containing the coordinates of the plotted points (NULL when not plotted).

References

For RC(M) models:

Goodman, L.A. (1991). Measures, Models, and Graphical Displays in the Analysis of Cross-Classified Data. *J. of the Am. Stat. Association* 86(416), 1085-1111.

Clogg, C.C., and Shihadeh, E.S. (1994). Statistical Models for Ordinal Variables. Sage: Advanced Quantitative Techniques in the Social Sciences (4).

Wong, R.S-K. (2010). Association models. Sage: Quantitative Applications in the Social Sciences (164).

For van der Heijden & Mooijaart models:

van der Heijden, P.G.M., and Mooijaart, A. (1995). Some new log bilinear models for the analysis of asymmetry in a square contingency table. *Sociol. Methods and Research* 24, 7-29.

See Also

rc, rcL, rcL.trans, hmskew, hmskewL, yrcskew

Examples

```
## Wong (2010), Figures 2.2 and 2.3 (p. 50-51)
data(gss8590)

## Not run:
model <- rc(margin.table(gss8590[,,c(2,4)], 1:2),
            nd=2, weighting="none", se="jackknife")
plot(model, what="row", rev.axes=c(TRUE, FALSE), conf.int=0.95)
plot(model, what="col", rev.axes=c(TRUE, FALSE), conf.int=0.95)

## End(Not run)

## Wong (2010), Figures 4.1 and 4.2 (p. 108-109)
data(gss7590)
model <- rcL(gss7590, nd=2, weighting="none")
```

```
opar <- par(mfrow=c(2, 2))
for(i in 1:4)
    plot(model, layer=i, what="rows", rev.axes=c(TRUE, FALSE),
         main=rownames(model$assoc$phi)[i],
         xlim=c(-1.2, 1.2), ylim=c(-1.2, 1.2))

par(mfrow=c(2, 2))
for(i in 1:4)
    plot(model, layer=i, what="col", rev.axes=c(TRUE, FALSE),
         main=rownames(model$assoc$phi)[i],
         xlim=c(-1.4, 1.4), ylim=c(-1.2, 1.2))

par(opar)


## van der Heijden & Mooijaart (1995), Figure 1c (p. 23)
data(ocg1973)
# 5:1 is here to take "Farmers" as reference category (angle 0)
model <- hmskew(ocg1973[5:1, 5:1], weighting="uniform")
# Reproduce the plot from the original article
plot(model, coords="cartesian")
# Use a polar coordinates system, which makes more sense in this setting
plot(model)
```

`plot.unidiff` *Plot Layer Coefficients From a UNIDIFF Model*

Description

Plots the layer coefficient estimates from a UNIDIFF model, together with confidence bars based on quasi-standard errors or "traditional" standard errors.

Usage

```
## S3 method for class 'unidiff'
plot(x, what = c("layer.coef", "phi", "maor"),
     se.type = c("quasi.se", "se"),
     conf.int = 0.95, numeric.auto = TRUE, type = "p",
     xlab = names(dimnames(x$data))[3], ylab = NULL,
     add = FALSE, ylim, ...)
```

Arguments

`x`	an object resulting from a call to `unidiff`
`what`	"layer.coefficient" to plot the layer coefficients in the log odds ratio scale, with a reference of 1 for the first layer; "phi" to plot the intrinsic association coefficient (on the log odds ratio scale); "maor" to plot the mean absolute odds ratio (see `maor`).

se.type	whether to use quasi-standard errors or "traditional" standard errors to compute confidence intervals.
conf.int	the confidence level to retain for confidence bars.
numeric.auto	whether layer names should be converted to numeric values when possible (see "Details" below).
type	what type of plot should be drawn: see plot. Set to "o" or "b" join points with lines.
xlab	a title for the x axis: see see title.
ylab	a title for the y axis: see see title; if NULL, an appropriate default is used.
add	whether to create a new plot using plot, or draw over the existing plot by calling points and segments directly.
ylim	the y limits of the plot.
...	Further arguments passed to plot.

Details

If numeric.auto = TRUE and layer names (issued from the dimnames of the third dimension of the original table) can be converted to numeric (i.e. they consist of figures), the position of points on the x axis will be determined by the value of the name. This makes most sense when layers represent years, especially when they are not regularly spaced. If this behaviour is disabled, layers will be placed regularly on the x axis, disregarding their possible interpretation as numeric values.

Author(s)

Milan Bouchet-Valat

See Also

unidiff, summary.unidiff

Examples

```
# See ?unidiff
```

---

ras *RAS/Deming-Stephan Algorithm for Raking Tables*

Description

Adjust a table by multiplying rows and columns in order to reproduce the provided margins, preserving all the odds ratios. This procedure is know as the RAS or Deming-Stephan algorithm, as iterative proportional fitting (IPF) or as biproportional fitting.

Usage

```
ras(tab, row, col, tolerance = .Machine$double.eps)
```

Arguments

tab	a two-way table with only positive or zero entries
row	a vector of strictly positive elements containing the wanted row margins (sums).
col	a vector of strictly positive elements containing the wanted column margins (sums).
tolerance	the convergence criterion to stop iterating.

Details

Note that `sum(row)` must be equal to `sum(col)` for the algorithm to make sense.

Value

The adjusted table with row sums equal to `row` and column sums equal to `col`.

Author(s)

Milan Bouchet-Valat

---

rc *Fitting Row-Column Association Models*

Description

Fit log-multiplicative row-column association models, also called RC(M) models or Goodman's (1979) Model II, with one or several dimensions. Supported variants (for square tables) include symmetric (homogeneous) row and column scores, possibly combined with separate diagonal parameters.

Usage

```
rc(tab, nd = 1, symmetric = FALSE, diagonal = FALSE,
   weighting = c("marginal", "uniform", "none"),
   rowsup = NULL, colsup = NULL,
   se = c("none", "jackknife", "bootstrap"),
   nreplicates = 100, ncpus = getOption("boot.ncpus"),
   family = poisson, weights = NULL,
   start = NULL, etastart = NULL, tolerance = 1e-8,
   iterMax = 5000, trace = FALSE, verbose = TRUE, ...)
```

Arguments

tab	a two-way table, or an object (such as a matrix) that can be coerced into a table; if present, dimensions above two will be collapsed.
nd	the number of dimensions to include in the model. Cannot exceed `min(nrow(tab) - 1, ncol(tab) - 1)` if `symmetric` is `FALSE` (saturated model), and twice this threshold otherwise (quasi-symmetry model).
symmetric	should row and column scores be constrained to be equal? Valid only for square tables.
diagonal	should the model include parameters specific to each diagonal cell? This amounts to taking quasi-independence, rather than independence, as the baseline model. Valid only for square tables.
weighting	what weights should be used when normalizing the scores.
rowsup	if present, a matrix with the same columns as `tab` giving supplementary (passive) rows. If `symmetric = TRUE`, `rowsup` and `colsup` must be specified together and rows of `rowsup` must correspond to columns of `colsup`.
colsup	if present, a matrix with the same rows as `tab` giving supplementary (passive) columns. See `rowsup`.
se	which method to use to compute standard errors for parameters (see `se.assoc`).
nreplicates	the number of bootstrap replicates, if enabled.
ncpus	the number of processes to use for jackknife or bootstrap parallel computing. Defaults to the number of cores (see `detectCores`), with a maximum of 5, but falls back to 1 (no parallelization) if package `parallel` is not available.
family	a specification of the error distribution and link function to be used in the model. This can be a character string naming a family function; a family function, or the result of a call to a family function. See `family` details of family functions.
weights	an optional vector of weights to be used in the fitting process.
start	either `NA` to use optimal starting values, `NULL` to use random starting values, or a vector of starting values for the parameters in the model.
etastart	starting values for the linear predictor; set to `NULL` to use either default starting values (if `start = NA`), or random starting values (in all other cases).
tolerance	a positive numeric value specifying the tolerance level for convergence; higher values will speed up the fitting process, but beware of numerical instability of estimated scores!
iterMax	a positive integer specifying the maximum number of main iterations to perform; consider raising this value if your model does not converge.
trace	a logical value indicating whether the deviance should be printed after each iteration.
verbose	a logical value indicating whether progress indicators should be printed, including a diagnostic error message if the algorithm restarts.
...	more arguments to be passed to `gnm`

Details

This function fits log-multiplicative row-column association models, usually called (after Goodman) RC(M) models, typically following the equation:

$$logF_{ij} = \lambda + \lambda_i^I + \lambda_j^J + \sum_{m=1}^{M} \phi_m \mu_{im} \nu_{jm}$$

where F_{ij} is the expected frequency for the cell at the intersection of row i and column j of `tab`, and M the number of dimensions. See references for detailed information about the variants of the model, the degrees of freedom and the identification constraints applied to the scores.

Actual model fitting is performed using `gnm`, which implements the Newton-Raphson algorithm. This function simply ensures correct start values are used, in addition to allowing for identification of scores even with several dimensions, computation of their jackknife or bootstrap standard errors, and plotting. The default starting values for association parameters are computed using a singular/eigen value decomposition from the results of the model without association component ("base model"). In some complex cases, using `start = NULL` to start with random values can be more efficient, but it is also less stable and can converge to non-optimal solutions.

Value

A `rc` object, with all the components of a `gnm` object, plus an `assoc.rc` component holding the most relevant association information:

`phi`	The intrisic association parameters, one per dimension.
`row`	Row scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
`col`	Column scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
`weighting`	The name of the weighting method used, reflected by `row.weights` and `col.weights`.
`row.weights`	The row weights used for the identification of scores, as specified by the `weighting` argument.
`col.weights`	The column weights used for the identification of scores, as specified by the `weighting` argument.
`covmat`	The variance-covariance matrix for phi coefficients and normalized row and column scores. Only present if `se` was not "none".
`adj.covmats`	An array stacking on its third dimension one variance-covariance matrix for the adjusted scores of each layer in the model (used for plotting). Only present if `se` was not "none".
`covtype`	The method used to compute the variance-covariance matrix (corresponding to the `se` argument.

Author(s)

Milan Bouchet-Valat

References

Goodman, L.A. (1979). Simple Models for the Analysis of Association in Cross-Classifications having Ordered Categories. *J. of the Am. Stat. Association* 74(367), 537-552.

Becker, M.P., and Clogg, C.C. (1989). Analysis of Sets of Two-Way Contingency Tables Using Association Models. *Journal of the American Statistical Association* 84(405), 142-151.

Goodman, L.A. (1985). The Analysis of Cross-Classified Data Having Ordered and/or Unordered Categories: Association Models, Correlation Models, and Asymmetry Models for Contingency Tables With or Without Missing Entries. *The Annals of Statistics* 13(1), 10-69.

Goodman, L.A. (1991). Measures, Models, and Graphical Displays in the Analysis of Cross-Classified Data. *J. of the Am. Stat. Association* 86(416), 1085-1111.

Clogg, C.C., and Shihadeh, E.S. (1994). Statistical Models for Ordinal Variables. Sage: Advanced Quantitative Techniques in the Social Sciences (4).

Wong, R.S-K. (2010). Association models. SAGE: Quantitative Applications in the Social Sciences.

See Also

plot.rc, gnm

Examples

```
## Goodman (1991), Table 17.1 (p. 1097)
data(criminal)
model <- rc(criminal)

model$assoc # These are the phi (.07), mu and nu
model$assoc$row[,1,1] * model$assoc$phi[1,1] # These are the mu'
model$assoc$col[,1,1] * model$assoc$phi[1,1] # These are the nu'

## Becker & Clogg (1989), Table 5 (p. 145)
# See also ?rcL to run all models in one call
## Not run:
data(color)

# "Uniform weights" in the authors' terms mean "no weighting" for us
# See ?rcL for average marginals
caithness.unweighted <- rc(color[,,1], nd=2, weighting="none",
                           se="jackknife")
caithness.marginal <- rc(color[,,1], nd=2, weighting="marginal",
                         se="jackknife")
aberdeen.unweighted <- rc(color[,,2], nd=2, weighting="none",
                          se="jackknife")
aberdeen.marginal <- rc(color[,,2], nd=2, weighting="marginal",
                        se="jackknife")

caithness.unweighted
caithness.marginal
aberdeen.unweighted
aberdeen.marginal
```

```
  # To see standard errors, either:
  se(caithness.unweighted)

  # and so on...
  # (ours are much smaller for the marginal-weighted case)
  # Or:
  summary(caithness.unweighted)

## End(Not run)


  ## Clogg & Shihadeh (1994), Tables 3.5a and b (p. 55-61)
  data(gss88)
  model <- rc(gss88)

  # Unweighted scores
  summary(model, weighting="none")
  # Marginally weighted scores
  summary(model, weighting="marginal")
  # Uniformly weighted scores
  summary(model, weighting="uniform")


  ## Wong (2010), Table 2.7 (p. 48-49)
  ## Not run:
  data(gss8590)

  # The table used in Wong (2001) is not perfectly consistent
  # with that of Wong (2010)
  tab <- margin.table(gss8590[,,c(2,4)], 1:2)
  tab[2,4] <- 49

  model <- rc(tab, nd=2, weighting="none", se="jackknife")

  model
  summary(model) # Jackknife standard errors are slightly different
                 # from their asymptotic counterparts

  # Compare with bootstrap standard errors
  model2 <- rc(tab, nd=2, weighting="none", se="bootstrap")
  plot(model, conf.int=0.95)
  summary(model2)

## End(Not run)
```

---

rcL *Fitting Row-Column Association Models With Layer Effect*

Description

Fit log-multiplicative row-column association models with layer effect, also called RC(M)-L models, with one or several dimensions. Supported variants include homogeneous or heterogeneous scores over the layer variable, and (for square tables) symmetric (homogeneous) row and column scores, possibly combined with separate diagonal parameters.

Usage

```
rcL(tab, nd = 1,
    layer.effect = c("homogeneous.scores", "heterogeneous", "none"),
    symmetric = FALSE,
    diagonal = c("none", "heterogeneous", "homogeneous"),
    weighting = c("marginal", "uniform", "none"),
    se = c("none", "jackknife", "bootstrap"),
    nreplicates = 100, ncpus = getOption("boot.ncpus"),
    family = poisson, weights = NULL,
    start = NULL, etastart = NULL, tolerance = 1e-8,
    iterMax = 5000, eliminate=NULL,
    trace = FALSE, verbose = TRUE, ...)
```

Arguments

`tab`	a three-way table, or an object (such as a matrix) that can be coerced into a table; if present, dimensions above three will be collapsed.
`nd`	the number of dimensions to include in the model. Cannot exceed `min(nrow(tab) - 1, ncol(tab) - 1)` if `symmetric` is `FALSE` (saturated model), and twice this threshold otherwise (quasi-symmetry model).
`layer.effect`	determines the form of the interaction between row-column association and layers. See "Details" below.
`symmetric`	should row and column scores be constrained to be equal? Valid only for square tables.
`diagonal`	what type of diagonal-specific parameters to include in the model, if any. This amounts to taking quasi-conditional independence, rather than conditional independence, as the baseline model. Valid only for square tables.
`weighting`	what weights should be used when normalizing the scores.
`se`	which method to use to compute standard errors for parameters.
`nreplicates`	the number of bootstrap replicates, if enabled.
`ncpus`	the number of processes to use for jackknife or bootstrap parallel computing. Defaults to the number of cores (see `detectCores`), with a maximum of 5, but falls back to 1 (no parallelization) if package `parallel` is not available.
`family`	a specification of the error distribution and link function to be used in the model. This can be a character string naming a family function; a family function, or the result of a call to a family function. See `family` details of family functions.
`weights`	an optional vector of weights to be used in the fitting process.

start	either NA to use optimal starting values, NULL to use random starting values, or a vector of starting values for the parameters in the model.
etastart	starting values for the linear predictor; set to NULL to use either default starting values (if start = NA), or random starting values (in all other cases).
tolerance	a positive numeric value specifying the tolerance level for convergence; higher values will speed up the fitting process, but beware of numerical instability of estimated scores!
iterMax	a positive integer specifying the maximum number of main iterations to perform; consider raising this value if your model does not converge.
eliminate	either NULL (the default) to estimate all parameters, NA to skip the estimation of some parameters for increased efficiency, or the name of a factor to be passed as gnm's corresponding argument.
trace	a logical value indicating whether the deviance should be printed after each iteration.
verbose	a logical value indicating whether progress indicators should be printed, including a diagnostic error message if the algorithm restarts.
...	more arguments to be passed to gnm

Details

This function fits log-multiplicative row-column association models with layer effect, usually called (after Wong) RC(M)-L models, typically following the equation:

$$logF_{ijk} = \lambda + \lambda_i^I + \lambda_j^J + \lambda_k^K + \lambda_{ik}^{IK} + \lambda_{jk}^{JK} + \sum_{m=1}^{M} \phi_{mk}\mu_{imk}\nu_{jmk}$$

where F_{ijk} is the expected frequency for the cell at the intersection of row i, column j and layer k of tab, and M the number of dimensions. If layer.effect is set to 'heterogeneous', different scores will be computed for each level, which is equivalent to fitting separate RC(M) models on the k two-way tables. If it is set to 'homogeneous.scores', then $\mu_{imk} = \mu_{mk}$ and $\nu_{imk} = \nu_{im}$ for all layers k: only the ϕ_{mk} are allowed to vary across layers. If it is set to 'none', then in addition to the previous conditions all ϕ_{mk} are forced to be equal for all layers k, which amounts to a stability of the association across layers. See references for detailed information about the variants of the model, the degrees of freedom and the identification constraints applied to the scores.

Actual model fitting is performed using gnm, which implements the Newton-Raphson algorithm. This function simply ensures correct start values are used, in addition to allowing for identification of scores even with several dimensions, computation of their jackknife or bootstrap standard errors, and plotting. The default starting values for association parameters are computed using a singular/eigen value decomposition from the results of the model without association component ("base model"). In some complex cases, using start = NULL to start with random values can be more efficient, but it is also less stable and can converge to non-optimal solutions.

Value

A rcL object, with all the components of a gnm object, plus an assoc component holding the most relevant association information:

phi	The intrisic association parameters, one per dimension and per layer.
row	Row scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
col	Column scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
weighting	The name of the weighting method used, reflected by row.weights and col.weights.
row.weights	The row weights used for the identification of scores, as specified by the weighting argument.
col.weights	The column weights used for the identification of scores, as specified by the weighting argument.
covmat	The variance-covariance matrix for phi coefficients and normalized row and column scores. Only present if se was not "none".
adj.covmats	An array stacking on its third dimension one variance-covariance matrix for the adjusted scores of each layer in the model (used for plotting). Only present if se was not "none".
covtype	The method used to compute the variance-covariance matrix (corresponding to the se argument.

Author(s)

Milan Bouchet-Valat

References

Wong, R.S-K. (2010). Association models. SAGE: Quantitative Applications in the Social Sciences.

See Also

plot.rcL, gnm

Examples

```
## Becker & Clogg (1989), Table 5 (p. 145)
# See also ?rc for more details
## Not run:
data(color)

# "Uniform weights" in the authors' terms mean "no weighting" for us,
# and "average marginals" means "marginal" with rcL
# See ?rc for "marginals"
unweighted <- rcL(color, nd=2, weighting="none",
                  layer.effect="heterogeneous", se="jackknife")
marginal <- rcL(color, nd=2, weighting="marginal",
                layer.effect="heterogeneous", se="jackknife")
unweighted
marginal
```

```
  # (our standard errors are much smaller for the marginal-weighted case)
  summary(unweighted)
  summary(marginal)

  opar <- par(mfrow=c(1, 2))
  plot(marginal, layer="Caithness", conf.int=0.95)
  plot(marginal, layer="Aberdeen", conf.int=0.95)
  par(opar)

## End(Not run)


  ## Wong (2010), Table 4.6 (p. 103), model 9
  ## Not run:
  data(gss7590)

  model <- rcL(gss7590, nd=2, weighting="none", se="jackknife")

  model
  summary(model) # Jackknife standard errors are slightly different
                 # from their asymptotic counterparts

  # See ?plot.rcL for plotting

## End(Not run)
```

`rcL.trans` *Fitting Row-Column Association Models With Transitional Layer Effect*

Description

Fit log-multiplicative row-column association models with transitional layer effect, which are related to the RC(M)-L model, with one or several dimensions. Supported variants include (for square tables) symmetric (homogeneous) row and column scores, possibly combined with separate diagonal parameters.

Usage

```
rcL.trans(tab, nd = 1,
          symmetric = FALSE,
          diagonal = c("none", "heterogeneous", "homogeneous"),
          weighting = c("marginal", "uniform", "none"),
          se = c("none", "jackknife", "bootstrap"),
          nreplicates = 100, ncpus = getOption("boot.ncpus"),
          family = poisson, weights = NULL,
          start = NULL, etastart = NULL, tolerance = 1e-8,
          iterMax = 5000, trace = FALSE, verbose = TRUE, ...)
```

Arguments

tab	a three-way table, or an object (such as a matrix) that can be coerced into a table; if present, dimensions above three will be collapsed.
nd	the number of dimensions to include in the model. Cannot exceed `min(nrow(tab) - 1, ncol(tab) - 1)` if `symmetric` is `FALSE` (saturated model), and twice this threshold otherwise (quasi-symmetry model).
symmetric	should row and column scores be constrained to be equal? Valid only for square tables.
diagonal	what type of diagonal-specific parameters to include in the model, if any. This amounts to taking quasi-conditional independence, rather than conditional independence, as the baseline model. Valid only for square tables.
weighting	what weights should be used when normalizing the scores.
se	which method to use to compute standard errors for parameters.
nreplicates	the number of bootstrap replicates, if enabled.
ncpus	the number of processes to use for jackknife or bootstrap parallel computing. Defaults to the number of cores (see `detectCores`), with a maximum of 5, but falls back to 1 (no parallelization) if package `parallel` is not available.
family	a specification of the error distribution and link function to be used in the model. This can be a character string naming a family function; a family function, or the result of a call to a family function. See `family` details of family functions.
weights	an optional vector of weights to be used in the fitting process.
start	either `NA` to use optimal starting values, `NULL` to use random starting values, or a vector of starting values for the parameters in the model.
etastart	starting values for the linear predictor; set to `NULL` to use either default starting values (if `start = NA`), or random starting values (in all other cases).
tolerance	a positive numeric value specifying the tolerance level for convergence; higher values will speed up the fitting process, but beware of numerical instability of estimated scores!
iterMax	a positive integer specifying the maximum number of main iterations to perform; consider raising this value if your model does not converge.
trace	a logical value indicating whether the deviance should be printed after each iteration.
verbose	a logical value indicating whether progress indicators should be printed, including a diagnostic error message if the algorithm restarts.
...	more arguments to be passed to `gnm`

Details

This function fits log-multiplicative row-column association models with regression-type layer effect which are **experimental** models combining the principles behind RC(M)-L (Wong, 2010; see `rcL`) and regression-type models (Goodman & Hout, 1998). More specifically, like RC(M)-L models, row and column scores are allowed to vary across a layer variable, and the pattern of this variation follows the regression-type inspiration: for each dimension, a set of scores describes the

first layer, another set describes the total variation of these scores need to describe the association observed for the last layer, and one parameter per layer describes the position of the layer between the first and the last layer. Compared with the RC(M)-L model with homogeneous scores across layers, this models allows for a finer description of changes since the ordering and distances of categories on a dimension are allowed to vary, and not only the general strength of the association. It is designed to describe transitions from one state to another, and is best suited for ordered layer variables like time (though the model is not sensitive to reordering of the layers).

The general equation of the model is:

$$logF_{ijk} = \lambda + \lambda_i^I + \lambda_j^J + \lambda_k^K + \lambda_{ik}^{IK} + \lambda_{jk}^{JK} + \sum_{m=1}^{M} \phi_{mk}(\mu_{im}^S + \psi_{mk}\mu_{im}^V)(\nu_{jm}^S + \psi_{mk}\nu_{jm}^V)$$

where F_{ijk} is the expected frequency for the cell at the intersection of row i, column j and layer k of `tab`, and M the number of dimensions. The ψ_{mk} parameter is constrained to be positive, equal to 0 for the first layer ($m = 1$), and equal to 1 for the last layer.

This model should not be confused with another combination of RC(M) models with the regression-type approach, presented by Goodman & Hout (1998:180), in which two separate RC(M) associations are used to describe respectively the stable and the varying components. In the present model, row and column scores for both components are summed *before* entering the multiplicative interaction, which means only one RC(M) association exists.

The returned object is a generic `rcL` association model describing the fitted scores for each layer. To analyze more specifically the variation of each (normalized) score from the first to the last layer, use: `model$assoc$row[,dim(model$assoc$row)[3]] - model$assoc$row[,1]` (and similarly for column scores).

Actual model fitting is performed using `gnm`, which implements the Newton-Raphson algorithm. This function simply ensures correct start values are used, in addition to allowing for identification of scores even with several dimensions, computation of their jackknife or bootstrap standard errors, and plotting. The default starting values are taken from a model with a stable RC(M) association ("base model"). In some complex cases, using `start = NULL` to get random starting values can be more efficient, but it is also less stable and can converge to non-optimal solutions.

Value

A `rcL` object, with all the components of a `gnm` object, plus an `assoc` component holding the most relevant association information:

`phi`	The intrisic association parameters, one per dimension and per layer.
`row`	Row scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
`col`	Column scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
`weighting`	The name of the weighting method used, reflected by `row.weights` and `col.weights`.
`row.weights`	The row weights used for the identification of scores, as specified by the `weighting` argument.
`col.weights`	The column weights used for the identification of scores, as specified by the `weighting` argument.

covmat	The variance-covariance matrix for phi coefficients and normalized row and column scores. Only present if se was not "none".
adj.covmats	An array stacking on its third dimension one variance-covariance matrix for the adjusted scores of each layer in the model (used for plotting). Only present if se was not "none".
covtype	The method used to compute the variance-covariance matrix (corresponding to the se argument.

Author(s)

Milan Bouchet-Valat

References

Goodman, L.A., and Hout, M. (1998). Statistical Methods and Graphical Displays for Analyzing How the Association Between Two Qualitative Variables Differs Among Countries, Among Groups, Or Over Time: A Modified Regression-Type Approach. *Sociological Methodology* 28(1), 175-230. Wong, R.S-K. (2010). Association models. SAGE: Quantitative Applications in the Social Sciences.

See Also

plot.rcL, gnm

RCTrans *Specify a Row-Column Association With Transitional Layer Effect in a gnm Model Formula*

Description

A function of class "nonlin" to specify a log-multiplicative row-column association models with transitional layer effect with one or several dimensions in the formula argument to gnm. RCTransSymm allows specifying a variant with symmetric (homogeneous) row and column scores.

Usage

```
RCTrans(row, col, layer, inst = NULL)
RCTransSymm(row, col, layer, inst = NULL)
```

Arguments

row	the levels of the row variable
col	the levels of the column variable
layer	the levels of the layer variable
inst	a positive integer specifying the instance number of the term

Details

This function is used by `rcL.trans` to fit an experimental model.

`RCTrans` combines its arguments in the following way:

$$RCTrans(i, j, k) = (\mu_i^S + \psi_k \mu_i^V)(\nu_j^S + \psi_k \nu_j^V)$$

where $RCTrans(i, j, k)$ is the skew association for the cell at the intersection of row i, column j and layer k of the table.

`RCTransSymm` is similar, but forces μ_i^S and ν_i^S (respectively μ_i^V and ν_i^V) to be equal for identical values of i (diagonal cells).

Value

A list with the required components of a `"nonlin"` function:

`predictors`	the expressions passed to `Mult`
`term`	a function to create a deparsed mathematical expression of the term, given labels for the predictors.
`call`	the call to use as a prefix for parameter labels.

Author(s)

Milan Bouchet-Valat

See Also

`rcL.trans`

---

se.assoc *Standard Errors for Association Models*

Description

Get standard errors for log-multiplicative association scores and intrinsic association coefficients.

Usage

```
se(x, ...)

## S3 method for class 'assoc'
se(x, type = c("se", "quasi.se"), ...)

## S3 method for class 'rc'
se(x, type = c("se", "quasi.se"), ...)

## S3 method for class 'hmskew'
se(x, type = c("se", "quasi.se"), ...)
```

```
## S3 method for class 'yrcskew'
se(x, type = c("se", "quasi.se"), ...)

## S3 method for class 'rcL'
se(x, type = c("se", "quasi.se"), ...)
```

Arguments

x	an assoc object with a non-null covmat component (for se.assoc);. or a rc, hmskew, hmskewL, yrcskew, rcL or rcL.trans object fitted with the se argument different from "none" (for other functions).
type	the type of standard errors to be computed (see "Details" below).
...	currently unused.

Details

Currently, only jackknife or bootstrap standard errors are supported, depending on the se argument passed when fitting the model. **Some care is needed before using such standard errors and confidence intervals**. First one must ensure all model replicates converged to a correct solution, especially for bootstrap; second, when relying on normal confidence intervals computed from these standard errors, one must ensure that the coefficients estimators follow a normal distribution. Both checks can be performed by calling plot.boot on the boot.results component of the assoc object of the models (not supported for jackknife), with the index argument identifying the coefficient of interest (call colnames on the t member of the boot.results object to find out the index you need).

If outliers are present, standard errors and confidence intervals will be artificially large; to fix this, the tolerance argument must be set to a smaller value when fitting the models (which may in turn require increasing the value of the iterMax argument if convergence is too slow). Once outliers are removed, if coefficient estimates are still not normally distributed, robust bootstrap confidence intervals can be computed using boot.ci on the same object, provided a large number of replicates (> 1000) were computed.

For each replicate, stable scores and intrinsic association coefficients are identified using an orthogonal Procrustes analysis to suppress meaningless variations due to random reflections, permutations and rotations of dimensions (Milan & Whittaker, 1995). For hmskew and hmskewL models, a rotation within each pair of dimensions and a permutation of pairs of dimensions is performed, but no reflection as it would change the sign of intrinsic association coefficients.

Quasi-standard errors are computed using qvcalc. See the help page for this function for details and references about them.

Value

An object of the same form as the assoc component of the model, but with standard errors rather than the corresponding coefficients.

Author(s)

Milan Bouchet-Valat

References

Milan, L., and J. Whittaker (1995). Application of the Parametric Bootstrap to Models that Incorporate a Singular Value Decomposition. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 44(1), 31-49.

See Also

assoc, rc, hmskew, hmskewL, yrcskew, rcL, rcL.trans

Examples

```
# See ?rc about Wong (2010)
```

`summary.anoas` *Summary and Print Methods for ANOAS objects*

Description

These functions print the summary of a list of models fitted using the anoas function.

Usage

```
## S3 method for class 'anoas'
summary(object, ...)

## S3 method for class 'anoas'
print(x, ...)

## S3 method for class 'summary.anoas'
print(x, digits = 1, nsmall = 2, scientific = FALSE, ...)
```

Arguments

`object`	an anoas object.
`x`	an anoas object.
`digits`	See ?format.
`nsmall`	See ?format.
`scientific`	See ?format.
`...`	more arguments to be passed to further methods (ignored by `summary.anoas`).

Details

Contrary to most analyses of association in the literature, this function currently does not fit uniform association model ("U"), nor separate models with only row and column association ("R" and "C" models), nor log-linear row and column association models.

Currently, no significance test is performed on the models. Please note that it is not correct to test the one-dimension association model against the independence model.

Value

A `data.frame` with the following columns:

`Res. Df`	the residual number of degrees of freedom of the model.
`Res. Dev`	the residual deviance of the model (likelihood ratio Chi-squared statistic, or L-squared).
`Dev. Indep. (%)`	the ratio of the residual deviance of the model over that of the independence model, times 100. This measures the share of departure from independence that cannot be explained using the number of dimensions of the model.
`Dissim. (%)`	the dissimilarity index of the model's fitted values with regard to the observed data.
`BIC`	the Bayesian Information Criterion for the model.
`AIC`	Akaike's An Information Criterion for the model.
`Deviance`	the reduction in deviance of the model compared to the previous one
`Df`	the reduction in the number of degrees of freedom of the model compared to the previous one.

Author(s)

Milan Bouchet-Valat

See Also

anoas, anoasL

`summary.assoc` *Summarize Association Model Fits*

Description

`summary` method for objects of class `assocmod`, including rc, rcL, rcL.trans, hmskew, hmskewL and yrcskew models.

Usage

```
## S3 method for class 'assocmod'
summary(object, weighting, ...)

## S3 method for class 'summary.assocmod'
print(x, digits = max(3, getOption("digits") - 4), ...)
```

Arguments

object	an association model of class assocmod.
x	an object of class summary.gnm.
weighting	what weights should be used when normalizing the scores.
digits	the number of siginificant digits to use when printing.
...	further arguments passed to printCoefmat by print.summary.assocmod, and currently ignored by summary.assocmod.

Details

print.summary.assocmod prints the original call to assoc; a summary of the deviance residuals from the model fit; the coefficients of interest of the model; the residual deviance; the residual degrees of freedom; the Schwartz's Bayesian Information Criterion value; the Akaike's An Information Criterion value.

Association coefficients are printed with their standard errors, p-values and significance stars. The "Normalized" columns contains *normalized scores*, i.e. their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null. For models with only one layer (rc, hmskew, yrcskew), *adjusted* scores are printed in the "Adjusted" column: these correspond to normalized scores times the square root of the corresponding intrinsic association parameter (phi).

p-values correspond to normalized scores, and are computed under the assumption that estimators of coefficients are normally distributed, even if jackknife of bootstrap are used. See se.assoc for details about checking this assumption and the validity of jackknife or bootstrap results.

Note that setting the weighting argument to a value different from that used at the time of the fit discards the computed standard errors, if any.

Value

An object of class summary.assoc, with the following components:

call	the call component from object.
diagonal	the diagonal component from the object's assoc component.
deviance.resid	the deviance residuals, see residuals.glm.
coefficients	a matrix holding the association coefficients estimates, standard errors and p-values.
diagonal	a matrix holding the diagonal coefficients, if any.
weighting	the weigthing method used when normalizing the scores.
deviance	the deviance component from object.
chisq	the Pearson Chi-squared statistic for the model fit.
dissim	the dissimilarity index for the model fit.
df.residual	the df.residual component from object.
bic	the value of the BIC for the model fit (contrary to the value reported by AIC and extractAIC, the reference is 0 for the saturated model).

aic	the value of the AIC for the model fit (contrary to the value reported by AIC and extractAIC, the reference is 0 for the saturated model).
family	the family component from object.
dispersion	the estimated dispersion
df	a 3-vector of the rank of the model; the number of residual degrees of freedom; and number of unconstrained coefficients.

Author(s)

Milan Bouchet-Valat

See Also

assoc, plot.assoc, rc, rcL, rcL.trans, hmskew, hmskewL, yrcskew

---

summary.unidiff *Summarize UNIDIFF Model Fits*

Description

summary method for objects of class unidiff.

Usage

```
## S3 method for class 'unidiff'
summary(object, ...)

## S3 method for class 'summary.unidiff'
print(x, digits = max(3, getOption("digits") - 4), ...)
```

Arguments

object	an object resulting from a call to unidiff
x	an object of class summary.gnm.
digits	the number of siginificant digits to use when printing.
...	further arguments passed to printCoefmat by print.summary.unidiff, and currently ignored by summary.unidiff.

Details

print.summary.unidiff prints the original call to unidiff; a summary of the deviance residuals from the model fit; the coefficients of interest of the model; the residual deviance; the residual degrees of freedom; the Schwartz's Bayesian Information Criterion value; the Akaike's An Information Criterion value.

Layer and two-way interaction coefficients are printed with their standard errors, quasi-standard errors (see qvcalc), p-values (based on standard errors) and significance stars. Constrained coefficients have a value of 0 (by default), and 0 standard errors, but still have quasi-standard errors.

Value

An object of class `summary.unidiff`, with the following components:

`call`	the `call` component from object.
`deviance.resid`	the deviance residuals, see `residuals.glm`.
`layer`	a `data.frame` holding the layer coefficients estimates, standard errors, quasi-standard errors (see `qvcalc`) and p-values.
`phi.layer`	a `data.frame` holding the layer coefficients estimates, standard errors, and quasi-standard errors (see `qvcalc`) multiplied by the intrinsic association coefficient (see `maor`) for the first layer; p-values are the same as those for the "layer" component.
`interaction`	a `data.frame` holding the two-way interaction coefficients estimates, standard errors and p-values.
`deviance`	the `deviance` component from object.
`diagonal`	the `diagonal` component from the object's `unidiff` component.
`weighting`	the `weighting` component from the object's `unidiff` component.
`chisq`	the Pearson Chi-squared statistic for the model fit.
`dissim`	the dissimilarity index for the model fit.
`df.residual`	the `df.residual` component from object.
`bic`	the value of the BIC for the model fit (contrary to the value reported by `AIC` and `extractAIC`, the reference is 0 for the saturated model).
`aic`	the value of the AIC for the model fit (contrary to the value reported by `AIC` and `extractAIC`, the reference is 0 for the saturated model).
`family`	the `family` component from object.
`dispersion`	the estimated dispersion
`df`	a 3-vector of the rank of the model; the number of residual degrees of freedom; and number of unconstrained coefficients.

Author(s)

Milan Bouchet-Valat

See Also

`unidiff`, `plot.unidiff`

svyassocmod *Fitting Association Models With Complex Survey Data*

Description

Fit association models to data from a complex survey design, with inverse-probability weighting and (optionally) standard errors based on replicate weights.

Usage

```
svyrc(formula, design, nd = 1,
      symmetric = FALSE, diagonal = FALSE,
      weighting = c("marginal", "uniform", "none"),
      rowsup = NULL, colsup = NULL,
      Ntotal = nrow(design), exclude = c(NA, NaN),
      se = c("none", "replicate"),
      ncpus = getOption("boot.ncpus"),
      family = quasipoisson, weights = NULL,
      start = NULL, etastart = NULL, tolerance = 1e-8,
      iterMax = 5000, trace = FALSE, verbose = TRUE, ...)

svyhmskew(formula, design, nd.symm = NA, diagonal = FALSE,
          weighting = c("marginal", "uniform", "none"),
          rowsup = NULL, colsup = NULL,
          Ntotal = nrow(design), exclude = c(NA, NaN),
          se = c("none", "replicate"),
          ncpus = getOption("boot.ncpus"),
          family = quasipoisson, weights = NULL,
          start = NULL, etastart = NULL, tolerance = 1e-8,
          iterMax = 5000, trace = FALSE, verbose = TRUE, ...)

svyyrcskew(formula, design, nd.symm = NA, nd.skew = 1, diagonal = FALSE,
           weighting = c("marginal", "uniform", "none"),
           rowsup = NULL, colsup = NULL,
           Ntotal = nrow(design), exclude = c(NA, NaN),
           se = c("none", "replicate"),
           ncpus = getOption("boot.ncpus"),
           family = quasipoisson, weights = NULL,
           start = NA, etastart = NULL, tolerance = 1e-8,
           iterMax = 15000, trace = FALSE, verbose = TRUE, ...)

svyrcL(formula, design, nd = 1,
       layer.effect = c("homogeneous.scores",
                        "heterogeneous", "none"),
       symmetric = FALSE,
       diagonal = c("none", "heterogeneous", "homogeneous"),
       weighting = c("marginal", "uniform", "none"),
```

```
        Ntotal = nrow(design), exclude = c(NA, NaN),
        se = c("none", "replicate"),
        ncpus = getOption("boot.ncpus"),
        family = quasipoisson, weights = NULL,
        start = NULL, etastart = NULL, tolerance = 1e-8,
        iterMax = 5000, trace = FALSE, verbose = TRUE, ...)

svyrcL.trans(formula, design, nd = 1,
             symmetric = FALSE,
             diagonal = c("none", "heterogeneous", "homogeneous"),
             weighting = c("marginal", "uniform", "none"),
             Ntotal = nrow(design), exclude = c(NA, NaN),
             se = c("none", "replicate"),
             ncpus = getOption("boot.ncpus"),
             family = quasipoisson, weights = NULL,
             start = NULL, etastart = NULL, tolerance = 1e-8,
             iterMax = 5000, trace = FALSE, verbose = TRUE, ...)

svyhmskewL(formula, design, nd.symm = NA,
           layer.effect.skew = c("homogeneous.scores", "heterogeneous",
                                 "none"),
           layer.effect.symm = c("heterogeneous", "uniform",
                                 "homogeneous.scores", "none"),
           diagonal = c("none", "heterogeneous", "homogeneous"),
           weighting = c("marginal", "uniform", "none"),
           Ntotal = nrow(design), exclude = c(NA, NaN),
           se = c("none", "replicate"),
           ncpus = getOption("boot.ncpus"),
           family = quasipoisson, weights = NULL,
           start = NULL, etastart = NULL, tolerance = 1e-8,
           iterMax = 5000, trace = FALSE, verbose = TRUE, ...)
```

Arguments

`formula`	a formula specifying margins for the table (using '+' only) on which the model will be fitted (passed to `svytable`); dimensions of the resulting table must match the models expectations.
`design`	a survey object; if `se == "replicate"`, must be of class `svrepdesign` (see "Details" below).
`nd`	the number of dimensions to include in the model. Cannot exceed `min(nrow(tab) - 1, ncol(tab) - 1)` if `symmetric` is `FALSE` (saturated model), and twice this threshold otherwise (quasi-symmetry model).
`nd.symm`	the number of dimensions to include in the *symmetric* RC(M) association. Cannot exceed `2 * min(nrow(tab) - 1, ncol(tab) - 1)` (quasi-symmetry model). If `NA` (the default), a full quasi-symmetric association is used instead of a RC(M) model; if `0`, quasi-independence is used.
`nd.skew`	the number of dimensions to include in the *skew-symmetric* RC(M) association.

`layer.effect` determines the form of the interaction between row-column association and layers. See "Details" below.

`layer.effect.skew` determines the form of the interaction between skew-symmetric association and layers. See "Details" below.

`layer.effect.symm` determines the form of the interaction between symmetric row-column association, or quasi-symmetric association (if `nd.symm = NA`) and layers. See "Details" below.

`symmetric` should row and column scores be constrained to be equal? Valid only for square tables.

`diagonal` what type of diagonal-specific parameters to include in the model, if any. Only makes sense when `nd.symm` is not `NA` (else, diagonal parameters are already included).

`weighting` what weights should be used when normalizing the scores.

`Ntotal` sum of counts to normalize the table to (passed to `svytable`). See "Details" below..

`exclude` a vector of values to be exclude when building the table, passed to `xtabs`.

`rowsup` if present, a matrix with the same columns as `tab` and rows corresponding to the columns of `colsup`, giving supplementary (passive) rows.

`colsup` if present, a matrix with the same rows as `tab` and columns corresponding to the rows of `colsup`, giving supplementary (passive) columns.

`se` whether to compute replicate standard errors or not (only supported for `svrepdesign` objects).

`ncpus` the number of processes to use for jackknife or bootstrap parallel computing. Defaults to the number of cores (see `detectCores`), with a maximum of 5, but falls back to 1 (no parallelization) if package `parallel` is not available.

`family` a specification of the error distribution and link function to be used in the model. This can be a character string naming a family function; a family function, or the result of a call to a family function. See `family` details of family functions.

`weights` an optional vector of weights to be used in the fitting process.

`start` either `NA` to use optimal starting values, `NULL` to use random starting values, or a vector of starting values for the parameters in the model.

`etastart` starting values for the linear predictor; set to `NULL` to use either default starting values (if `start = NA`), or random starting values (in all other cases).

`tolerance` a positive numeric value specifying the tolerance level for convergence; higher values will speed up the fitting process, but beware of numerical instability of estimated scores!

`iterMax` a positive integer specifying the maximum number of main iterations to perform; consider raising this value if your model does not converge.

`trace` a logical value indicating whether the deviance should be printed after each iteration.

`verbose` a logical value indicating whether progress indicators should be printed, including a diagnostic error message if the algorithm restarts.

`...` more arguments to be passed to `gnm`

Details

The model is fitted to a table with probabilities estimated by svytable and (when Ntotal = nrow(design)) with the sample size equal to the observed sample size, treating the resulting table as if it came from iid multinomial sampling, as described by Rao and Scott. This assumption affects the fit statistics but not parameter point estimates.

Standard errors that do not rely on this assumption can be computed by fitting the model using each series of replicate weights. If your data does not come with replicate weights, use as.svrepdesign to create them first, and pass the resulting svrepdesign object via the design argument.

Value

An assocmod object whose exact class depends on the function called.

Note

Note that printed fit statistics and degrees of freedom rely on the iid assumption. This is also the case of the variance-covariance matrix returned by the vcov.gnm function.

Author(s)

Milan Bouchet-Valat

References

Rao, J.N.K., Scott, A.J. (1984). On Chi-squared Tests For Multiway Contingency Tables with Proportions Estimated From Survey Data. *Annals of Statistics* 12, 46-60.

See Also

rc, hmskew, yrcskew, rcL, rcL.trans, hmskewL

svytable, svyloglin, svyglm, as.svrepdesign

---

svygnm *Fitting Generalized Nonlinear Models With Complex Survey Data*

Description

Fit association models to data from a complex survey design, with inverse-probability weighting and (optionally) standard errors based on replicate weights.

Usage

```
svygnm(formula, design, ...)
## S3 method for class 'svyrep.design'
svygnm(formula, design,
    subset = NULL, data.fun = NULL, rescale = NULL, rho = NULL,
    return.replicates = FALSE, keep.weights = FALSE, na.action,
    eliminate, ncpus = getOption("boot.ncpus"), ...)
```

Arguments

formula	a symbolic description of the nonlinear predictor.
design	a survey object; if se == "replicate", must be of class svrepdesign (see "Details" below). Must contain all variables in the formula
subset	expression to select a subpopulation
data.fun	function called on each replicate to generate the data argument passed to gnm. If not NULL, it will be passed design and ... as arguments, and must return a data.frame object. This is primarily useful to compute a frequency table and fit log-linear models.
rescale	Rescaling of weights, to improve numerical stability. The default rescales weights to sum to the sample size. Use FALSE to not rescale weights. For replicate-weight designs, use TRUE to rescale weights to sum to 1, as was the case before version 0.7.0.
rho	For replicate BRR designs, to specify the parameter for Fay's variance method, giving weights of rho and 2-rho
return.replicates	return the replicates as a component of the result?
keep.weights	whether to save the weights in the survey.design$pweights component of the result; note this typically uses a lot of memory.
na.action	handling of NAs
eliminate	a factor to be included as the first term in the model. gnm will exploit the structure of this factor to improve computational efficiency. See details.
ncpus	the number of CPU cores to use to run replicates. Pass NULL to use the actual number of cores with an upper limit of 5.
...	more arguments to be passed to gnm

Details

This function can be used in a similar way as svyglm, but for generalized nonlinear models. It computes standard errors using replicates only (i.e. no asymptotic standard errors). If your data does not come with replicate weights, use as.svrepdesign to create them first, and pass the resulting svrepdesign object via the design argument.

Value

An svygnm object.

Note

Note that printed fit statistics and degrees of freedom rely on the iid assumption. This is also the case of the variance-covariance matrix returned by the vcov.gnm function.

Author(s)

Milan Bouchet-Valat, based on the svyglm function by Thomas Lumley

References

Rao, J.N.K., Scott, A.J. (1984). On Chi-squared Tests For Multiway Contingency Tables with Proportions Estimated From Survey Data. *Annals of Statistics* 12, 46-60.

See Also

gnm, svyglm, as.svrepdesign

Examples

```
library(survey)
data(api)
dstrat<-svydesign(id=~1,strata=~stype, weights=~pw, data=apistrat, fpc=~fpc)
rstrat<-as.svrepdesign(dstrat)
glm.mod <- svyglm(api00~ell+meals+mobility, design=rstrat)
gnm.mod <- svygnm(api00~ell+meals+mobility, design=rstrat, ncpus=1)
# Both functions give the same result for GLMs
summary(glm.mod)
summary(gnm.mod)


# GNM, can only be fitted with svygnm()
summary(svygnm(api00~ell+meals+mobility, design=rstrat, family=poisson, ncpus=1))
```

---

unidiff *Fitting Log-Multiplicative Uniform Difference/Layer Effect Model*

Description

Fit the log-multiplicative uniform difference model (UNIDIFF, see Erikson & Goldthorpe, 1992), also called the log-multiplicative layer effect model (Xie, 1992). For square tables, diagonal cells can be handled separately.

Usage

```
unidiff(tab, diagonal = c("included", "excluded", "only"),
        constrain = "auto",
        weighting = c("marginal", "uniform", "none"), norm = 2,
        family = poisson,
        tolerance = 1e-8, iterMax = 5000, eliminate=NULL,
        trace = FALSE, verbose = TRUE,
        checkEstimability = TRUE, ...)
```

Arguments

tab	a three-way table, or an object (such as a matrix) that can be coerced into a table; if present, dimensions above three will be collapsed as appropriate.

`diagonal` `included` fits the standard model with full two-way interaction; `excluded` adds to this model diagonal-specific parameters for each years, effectively removing the influence of diagonal cells on the layer coefficients; `only` fits a model without the full two-way interaction, where only diagonal parameters are affected by the layer effect (see "Details" below).

`constrain` (non-eliminated) coefficients to constrain, specified by a regular expression, a numeric vector of indices, a logical vector, a character vector of names, or "[?]" to select from a Tk dialog. The default constrains to 0 the first layer parameter and interaction coefficients for the first row and column of the table.

`weighting` what weights should be used when normalizing coefficients. This does not affect layer coefficients, which are set to 1 for the first layer, but only two-way interaction coefficients and layer association levels, which are layer coefficients times the intrinsic association coefficient (see `maor`) for the first layer.

`norm` the norm to use to compute the mean absolute odds ratio (see `maor`).

`family` a specification of the error distribution and link function to be used in the model. This can be a character string naming a family function; a family function, or the result of a call to a family function. See `family` details of family functions.

`tolerance` a positive numeric value specifying the tolerance level for convergence; higher values will speed up the fitting process, but beware of numerical instability of estimated scores!

`iterMax` a positive integer specifying the maximum number of main iterations to perform; consider raising this value if your model does not converge.

`eliminate` either `NULL` (the default) to estimate all parameters, `NA` to skip the estimation of some parameters for increased efficiency, or the name of a factor to be passed as `gnm`'s corresponding argument.

`trace` a logical value indicating whether the deviance should be printed after each iteration.

`verbose` a logical value indicating whether progress indicators should be printed, including a diagnostic error message if the algorithm restarts.

`checkEstimability` a logical value indicating whether the estimability of the contrasts should be checked via `checkEstimable`. Disabling this check can improve performance for large models.

`...` more arguments to be passed to `gnm`

Details

The equation of the fitted model is:

$$logF_{ijk} = \lambda + \lambda_i^I + \lambda_j^J + \lambda_k^K + \lambda_{ik}^{IK} + \lambda_{jk}^{JK} + \phi_k \psi_{ij}^{IJ}$$

where F_{ijk} is the expected frequency for the cell at the intersection of row i, column j and layer k of `tab`. When `diagonal = "excluded"`, λ_{ijk}^{IJK} parameters are added but set to 0 when $i \neq j$ (off-diagonal). When `diagonal = "only"`, ψ_{ij}^{IJ} is set to 0 when $i \neq j$.

Note that by default `weighting="marginal"`, meaning that reported interaction coefficients do *not* correspond to what is usually expected in log-linear modeling. Use `weighting="none"` or `weighting="uniform"` to use more classic identification constraints (effects coding).

Layer coefficients ϕ_k are internally exponentiated in the gnm formula, which means the reported values are in log scale, with reference 0 for the first year. Interaction coefficients use the "sum" contrast, also known as "effect" coding, except when `diagonal` is different from `included`, in which case "treatment" constrast (a.k.a "reference" or "dummy" coding) is used.

Actual model fitting is performed using `gnm`, which implements the Newton-Raphson algorithm. This function simply allows for direct identification of the log-multiplicative parameters by setting the appropriate constraints, and improves performance by eliminating less interesting coefficients.

Value

A `unidiff` object, with all the components of a `gnm` object, plus an `unidiff` component holding the most relevant information:

`layer`	a `qvcalc` object holding the (log) layer coefficients, their standard errors and quasi-standard errors.
`phi`	the value of the intrinsic association coefficient (see `maor`) for each layer.
`maor`	the value of the Mean absolute odds ratio (see `maor`) for each layer.
`interaction`	a data frame object holding the two-way interaction coefficients, and their standard errors.
`diagonal`	the value of the `diagonal` argument above.
`weighting`	the value of the `weighting` argument above.

Author(s)

Milan Bouchet-Valat

References

Erikson, R., and Goldthorpe, J.H. (1992). The Constant Flux: A Study of Class Mobility in Industrial Societies. Oxford: Clarendon Press. Ch. 3.

Xie, Yu (1992). The Log-Multiplicative Layer Effect Model for Comparing Mobility Tables. *Am. Sociol. Rev.* 57(3):380-395.

Yaish, M. (1998). Opportunities, Little Change. Class Mobility in Israeli Society, 1974-1991. Ph.D. thesis, Nuffield College, University of Oxford.

Yaish, M. (2004). Class Mobility Trends in Israeli Society, 1974-1991. Lewiston: Edwin Mellen Press.

See Also

`plot.unidiff`, `summary.unidiff`

Examples

```
## Yaish (1998, 2004)
data(yaish)

# Last layer omitted because of low frequencies
```

```
yaish <- yaish[,,-7]

# Layer (education) must be the third dimension
yaish <- aperm(yaish, 3:1)

model <- unidiff(yaish)

model
summary(model)
plot(model)
```

YRCSkew *Specify a Skew-Symmetric Association in a gnm Model Formula*

Description

A function of class "nonlin" to specify a Yamaguchi (1990) skew-symmetric association in the formula argument to gnm.

Usage

```
YRCSkew(row, col, rowinf, rowsup, inst = NULL)
```

Arguments

row	for each cell in the table, the row category.
col	for each cell in the table, the column category.
rowinf	must be 1 for cells above the diagonal, 0 for cells below and on the diagonal.
rowsup	must be 1 for cells below the diagonal, 0 for cells above and on the diagonal.
inst	a positive integer specifying the instance number of the term.

Details

This function is used by yrcskew to fit the "row-column-effect skew-symmetric association (logbilinear) model with full quasi-symmetry (QS+RC_SK)" proposed by Yamaguchi (1990). It can be used directly to fit custom variants of the model not supported by yrcskew.

This function combines its arguments in the following way:

$$YRCSkew(row, col, rowinf, rowsup) = \delta_{rowinf} * \mu_{row} * (\mu_{col} - \mu_{row}) + \delta_{rowsup} * \nu_{col} * (\nu_{row} - \nu_{col})$$

When arguments are set according to what is suggested above, and the skew δ parameter is constrained to 1, this amounts to the equation:

$$YRCSkew_{ij} = \delta_{i<j}\nu_i(\nu_j - \nu_i) - \delta_{i>j}\nu_j(\nu_i - \nu_j) = (\delta_{i<j} - \delta_{i>j})\nu_{min(i,j)}(\nu_{max(i,j)} - \nu_{min(i,j)})$$

where $YRCSkew_{ij}$ is the skew association for the cell at the intersection of row i and column j of the table. See reference for mathematical details, and the code of yrcskew for real-world usage.

Value

A list with the required components of a "nonlin" function:

predictors	the expressions passed to Mult
term	a function to create a deparsed mathematical expression of the term, given labels for the predictors.
call	the call to use as a prefix for parameter labels.

Author(s)

Milan Bouchet-Valat

References

Yamaguchi, K. (1990). Some Models for the Analysis of Asymmetric Association in Square Contingency Tables with Ordered Categories. *Sociol. Methodology* 20, 181-212.

See Also

yrcskew

Examples

```
# See ?yrcskew.
```

---

yrcskew *Fitting Yamaguchi RC_SK Skew-Symmetric Association Model*

Description

Fit a skew-symmetric association model proposed in Yamaguchi (1990) to describe asymmetry of square tables. This model can be combined with symmetric association models like a quasi-symmetry (the default) or symmetric (homogeneous) RC(M) models.

Usage

```
yrcskew(tab, nd.symm = NA, nd.skew = 1, diagonal = FALSE,
        weighting = c("marginal", "uniform", "none"),
        se = c("none", "jackknife", "bootstrap"),
        nreplicates = 100, ncpus = getOption("boot.ncpus"),
        family = poisson, weights = NULL,
        start = NA, etastart = NULL, tolerance = 1e-8,
        iterMax = 15000, trace = FALSE, verbose = TRUE, ...)
```

Arguments

tab	a two-way table, or an object (such as a matrix) that can be coerced into a table; if present, dimensions above two will be collapsed.
nd.symm	the number of dimensions to include in the *symmetric* RC(M) association. Cannot exceed `2 * min(nrow(tab) - 1, ncol(tab) - 1)`. If NA, a quasi-symmetric association is used instead of a RC(M) model.
nd.skew	the number of dimensions to include in the *skew-symmetric* RC(M) association.
diagonal	should the model include parameters specific to each diagonal cell? This amounts to taking quasi-independence, rather than independence, as the baseline model.
weighting	what weights should be used when normalizing the scores.
se	which method to use to compute standard errors for parameters.
nreplicates	the number of bootstrap replicates, if enabled.
ncpus	the number of processes to use for jackknife or bootstrap parallel computing. Defaults to the number of cores (see detectCores), with a maximum of 5, but falls back to 1 (no parallelization) if package parallel is not available.
family	a specification of the error distribution and link function to be used in the model. This can be a character string naming a family function; a family function, or the result of a call to a family function. See family details of family functions.
weights	an optional vector of weights to be used in the fitting process.
start	either NA (the default) to use reasonable starting values, NULL to use random starting values, or a vector of starting values for the parameters in the model.
etastart	starting values for the linear predictor; set to NULL to use either default starting values (if start = NA), or random starting values (in all other cases).
tolerance	a positive numeric value specifying the tolerance level for convergence; higher values will speed up the fitting process, but beware of numerical instability of estimated scores!
iterMax	a positive integer specifying the maximum number of main iterations to perform; consider raising this value if your model does not converge.
trace	a logical value indicating whether the deviance should be printed after each iteration.
verbose	a logical value indicating whether progress indicators should be printed, including a diagnostic error message if the algorithm restarts.
...	more arguments to be passed to gnm

Details

The original presented by Yamaguchi (1990), called "row-column-effect skew-symmetric association (logbilinear) model with full quasi-symmetry (QS+RC_SK)", combines a skew-symmetric association with a quasi-symmetry baseline; it is the variant fitted by default by this function. If nd.symm is set to a positive integer value, though, variants using a RC(M) model to describe the *symmetric association* are used, with our without diagonal-specific parameters (depending on the value of the diagonal argument); among them is the HM_RC+RC_SK variant, when nd.symm is 1.

These models follow the equation:

$$logF_{ij} = q_{ij} + \delta_{i<j}\nu_i(\nu_j - \nu_i) - \delta_{i>j}\nu_j(\nu_i - \nu_j)$$

where F_{ij} is the expected frequency for the cell at the intersection of row i and column j of `tab`, and q_{ij} a quasi-symmetric or a RC(M) association. See reference for detailed information about the degrees of freedom and the identification constraints applied to the scores.

Please note that contrary to other association models, this model is sensitive to reorderings of rows and columns. You have to take care of passing a table whose categories follow a hierarchical order with a substantive meaning.

Another model presented in the paper, the homogeneous symmetric and skew-symmetric associations models (HM_(S+SK)) is not currently supported.

Actual model fitting is performed using `gnm`, which implements the Newton-Raphson algorithm. This function simply ensures correct start values are used, in addition to allowing for identification of scores even with several dimensions, computation of their jackknife or bootstrap standard errors, and plotting. The default starting values for main parameters are taken from the model without association parameters ("base model"); association parameters start with random starting values. In some complex cases, using `start = NULL` to get completely random starting values can be more efficient, but it is also less stable and can converge to non-optimal solutions.

Value

A `yrcskew` object, which is a subclass of an `rc.symm` object (see`rc`) unless `nd.symm` is `NA`. In addition to this class, it contains a `assoc.yrcskew` component holding information about the *skew-symmetric* association:

`phi`	The intrisic association parameters, one per dimension.
`row`	Row scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
`col`	Column scores, normalized so that their (weighted) sum is 0, their (weighted) sum of squares is 1, and their (weighted) cross-dimensional correlation is null.
`row.weights`	The row weights used for the identification of scores, as specified by the `weighting` argument.
`col.weights`	The column weights used for the identification of scores, as specified by the `weighting` argument.
`covmat`	The variance-covariance matrix for phi coefficients and normalized row and column scores. Only present if `se` was not "none".
`adj.covmats`	An array stacking on its third dimension one variance-covariance matrix for the adjusted scores of each layer in the model (used for plotting). Only present if `se` was not "none".
`covtype`	The method used to compute the variance-covariance matrix (corresponding to the `se` argument.

Warning

This family of model sometimes converges to a non-optimal solution, in which case the reported scores are wrong. To protect yourself from this problem, you are advised to run the models several times to find out which convergence point is the true one. Furthermore, when model converges slowly, restarting the fitting procedure may produce much better random starting values.

Author(s)

Milan Bouchet-Valat

References

Yamaguchi, K. (1990). Some Models for the Analysis of Asymmetric Association in Square Contingency Tables with Ordered Categories. *Sociol. Methodology* 20, 181-212.

See Also

plot.yrcskew, gnm

Examples

```
## Yamaguchi (1990), Table 5, p. 202, and Table 6B, p. 205
data(ocg1973)

# Simple symmetric RC(1) model ("Null skew-symmetry")
rc.model <- rc(ocg1973, diagonal=TRUE, symmetric=TRUE, weighting="none")
# Reported phi is slightly different, coefficients agree
rc.model

# Note model does not always converge, several attempts may be needed
# Here we set known starting values to be sure it works
set.seed(5)
model <- yrcskew(ocg1973, nd.symm=1, nd.skew=1, diagonal=TRUE, weighting="none")

# We do not get the same results as the author, but the smaller deviance
# indicates a better fit in our version (!)
model
```

Index