Package: CorrBin (via r-universe)

August 30, 2024

Title Nonparametrics with Clustered Binary and Multinomial Data

Version 1.6.2

Date 2024-08-28

Encoding UTF-8

Depends R(>= 2.6.0)

Imports boot, combinat, dirmult, mvtnorm

Suggests geepack, lattice

Description Implements non-parametric analyses for clustered binary and multinomial data. The elements of the cluster are assumed exchangeable, and identical joint distribution (also known as marginal compatibility, or reproducibility) is assumed for clusters of different sizes. A trend test based on stochastic ordering is implemented. Szabo A, George EO. (2010) <doi:10.1093/biomet/asp077>; George EO, Cheon K, Yuan Y, Szabo A (2016) <doi:10.1093/biomet/asw009>.

License GPL (>=2)

LazyLoad yes

RoxygenNote 7.2.3

Repository https://anikoszabo.r-universe.dev

RemoteUrl https://github.com/anikoszabo/corrbin

RemoteRef HEAD

RemoteSha 60d2fe7a850e59b2dfc26e715f49f4407564ce9e

Contents

`CorrBin-package` *Nonparametrics for Correlated Binary and Multinomial Data*

Description

This package implements nonparametric methods for analyzing exchangeable binary and multinomial data with variable cluster sizes with emphasis on trend testing. The input should specify the treatment group, cluster-size, and the number of responses (i.e. the number of cluster elements with the outcome of interest) for each cluster.

Details

- The `CBData`/`CMData` and `read.CBData`/`read.CMData` functions create a 'CBData' or 'CMData' object used by the analysis functions.
- `ran.CBData` and `ran.CMData` can be used to generate random binary or multinomial data using a variety of distributions.
- `mc.test.chisq` tests the assumption of marginal compatibility underlying all the methods, while `mc.est` estimates the distribution of the number of responses under marginal compatibility.
- Finally, `trend.test` performs three different tests for trend along the treatment groups for binomial data.

Author(s)

Aniko Szabo

Maintainer: Aniko Szabo <aszabo@mcw.edu>

References

Szabo A, George EO. (2009) On the Use of Stochastic Ordering to Test for Trend with Clustered Binary Data. *Biometrika*

Stefanescu, C. & Turnbull, B. W. (2003) Likelihood inference for exchangeable binary data with varying cluster sizes. *Biometrics*, 59, 18-24

Pang, Z. & Kuk, A. (2007) Test of marginal compatibility and smoothing methods for exchangeable binary data with unequal cluster sizes. *Biometrics*, 63, 218-227

`CBData` *Create a 'CBdata' object from a data frame.*

Description

The `CBData` function creates an object of class *CBData* that is used in further analyses. It identifies the variables that define treatment group, clustersize and the number of responses.

Usage

```
CBData(x, trt, clustersize, nresp, freq = NULL)
```

Arguments

`x`	a data frame with one row representing a cluster or potentially a set of clusters of the same size and number of responses
`trt`	the name of the variable that defines treatment group
`clustersize`	the name of the variable that defines cluster size
`nresp`	the name of the variable that defines the number of responses in the cluster
`freq`	the name of the variable that defines the number of clusters represented by the data row. If `NULL`, then each row is assumed to correspond to one cluster.

Value

A data frame with each row representing all the clusters with the same trt/size/number of responses, and standardized variable names:

`Trt`	factor, the treatment group
`ClusterSize`	numeric, the cluster size
`NResp`	numeric, the number of responses
`Freq`	numeric, number of clusters with the same values

Author(s)

Aniko Szabo

See Also

`read.CBData` for creating a `CBData` object directly from a file.

Examples

```
data(shelltox)
sh <- CBData(shelltox, trt="Trt", clustersize="ClusterSize", nresp="NResp")
str(sh)
```

CMData *Create a 'CMdata' object from a data frame.*

Description

The `CMData` function creates an object of class *CMData* that is used in further analyses. It identifies the variables that define treatment group, clustersize and the number of responses for each outcome type.

Usage

```
CMData(x, trt, nresp, clustersize = NULL, freq = NULL)
```

Arguments

`x`	a data frame with one row representing a cluster or potentially a set of clusters of the same size and number of responses for each outcome
`trt`	the name of the variable that defines treatment group
`nresp`	either a character vector with the names or a numeric vector with indices of the variables that define the number of responses in the cluster for each outcome type. If `clustersize` is `NULL`, then it will be calculated assuming that the `nresp` vector contains all the possible outcomes. If `clustersize` is given, then an additional category is created for the excess cluster members.
`clustersize`	the name or index of the variable that defines cluster size, or `NULL`. If `NULL`, its value will be calculated by adding the counts from the `nresp` variables. If defined, an additional response type will be created for the excess cluster members.
`freq`	the name or index of the variable that defines the number of clusters represented by the data row. If `NULL`, then each row is assumed to correspond to one cluster.

Value

A data frame with each row representing all the clusters with the same trt/size/number of responses, and standardized variable names:

`Trt`	factor, the treatment group
`ClusterSize`	numeric, the cluster size
`NResp.1-NResp.K+1`	numeric, the number of responses for each of the K+1 outcome types
`Freq`	numeric, number of clusters with the same values

Author(s)

Aniko Szabo

See Also

`read.CMData` for creating a `CMData` object directly from a file.

Examples

```
data(dehp)
dehp <- CMData(dehp, trt="Trt", nresp=c("NResp.1","NResp.2","NResp.3"))
str(dehp)
```

---

dehp *Developmental toxicology study of DEHP in mice*

Description

This data set is based on a National Toxicology Program study on diethylhexyl phthalate, DEHP. Pregnant CD-1 mice were randomly assigned to receive 0, 250, 500, 1000, or 1500 ppm of DEHP in their feed during gestational days 0-17. The uterine contents of the mice were examined for toxicity endpoints prior to normal delivery. The possible outcomes are 1) malformation, 2) death or resorption, 3) no adverse event.

Usage

```
data(dehp)
```

Format

A 'CMData' object, that is a data frame with the following variables

Trt	factor giving treatment group
ClusterSize	the size of the litter
NResp.1	the number of fetuses with a type 1 outcome (malformation)
NResp.2	the number of fetuses with a type 2 outcome (death or resorption)
NResp.3	the number of fetuses with a type 3 outcome (normal)
Freq	the number of litters with the given ClusterSize/NResp.1-NResp.3 combination

Source

National Toxicology Program, NTP Study TER84064.

References

Tyl, R. W., Price, C. J., Marr, M. C., and Kimmel, C. A. (1988). Developmental toxicity evaluation of dietary di(2-ethylhexy)phthalate in Fischer 344 rats and CD-1 mice. *Fundamental and Applied Toxicology* 10, 395-412.

Examples

```
data(dehp)
library(lattice)
pl <- xyplot(NResp.1/ClusterSize + NResp.2/ClusterSize + NResp.3/ClusterSize ~ Trt,
        data=dehp, outer=TRUE, type=c("p","a"), jitter.x=TRUE)
pl$condlevels[[1]] <- c("Malformed", "Dead", "Normal")
print(pl)
```

---

egde *EGDE data*

Description

The data set is based on a developmental toxicity experiment on the effect of ethylene glycol diethyl ether (EGDE) on fetal development of New Zealand white rabbits. In the study, four groups of pregnant does were randomly assigned to dose levels $0, 25, 50$, and 100 milligrams per kilogram body weight of EGDE. For each litter and at each dose level, the adverse response used is the combined number of fetal malformation and fetal death.

Usage

```
data(egde)
```

Format

A 'CBData' object, that is a data frame with the following variables

Trt	factor giving treatment group
ClusterSize	the size of the litter
NResp	the number of affected fetuses
Freq	the number of litters with the given ClusterSize/NResp combination

Source

Krewski, D., Zhu, Y., and Fung, K. (1995). Statistical analysis of overdispersed multinomial data from developmental toxicity studies. In *Statistics in Toxicology*, Ed. B. Morgan, pp.\ 151–179. New York: Oxford University Press.

Examples

```
data(egde)
stripchart(I(NResp/ClusterSize)~Trt, cex=sqrt(egde$Freq), data=egde, pch=1,
           method="jitter", vertical=TRUE, ylab="Proportion affected")
```

Extract *Extract from a CBData or CMData object*

Description

The extracting syntax works as for `[.data.frame`, and in general the returned object is not a `CBData` or `CMData` object. However if the columns are not modified, then the result is still a `CBData` or `CMData` object with appropriate attributes preserved, and the unused levels of treatment groups dropped.

Usage

```
## S3 method for class 'CBData'
x[i, j, drop]

## S3 method for class 'CMData'
x[i, j, drop]
```

Arguments

`x`	`CMData` object.
`i`	numeric, row index of extracted values
`j`	numeric, column index of extracted values
`drop`	logical. If TRUE the result is coerced to the lowest possible dimension. The default is the same as for `[.data.frame`: to drop if only one column is left, but not to drop if only one row is left.

Value

a CBData or CMData object

Author(s)

Aniko Szabo

See Also

CBData, CMData

Examples

```
data(shelltox)
str(shelltox[1:5,])
str(shelltox[1:5, 2:4])

data(dehp)
str(dehp[1:5,])
str(dehp[1:5, 2:4])
```

GEE.trend.test *GEE-based trend test*

Description

GEE.trend.test implements a GEE based test for linear increasing trend for correlated binary data.

Usage

```
GEE.trend.test(cbdata, scale.method = c("fixed", "trend", "all"))
```

Arguments

cbdata	a CBData object
scale.method	character string specifying the assumption about the change in the overdispersion (scale) parameter across the treatment groups: "fixed" - constant scale parameter (default); "trend" - linear trend for the log of the scale parameter; "all" - separate scale parameter for each group.

Details

The actual work is performed by the geese function of the geepack library, which is required for this feature to work. This function only provides a convenient wrapper to obtain the results in the same format as RS.trend.test and SO.trend.test.

The implementation aims for testing for *increasing* trend, and a one-sided p-value is reported. The test statistic is asymptotically normally distributed, and a two-sided p-value can be easily computed if needed.

Value

A list with components

`statistic`	numeric, the value of the test statistic
`p.val`	numeric, asymptotic one-sided p-value of the test

Author(s)

Aniko Szabo, aszabo@mcw.edu

See Also

`RS.trend.test`, `SO.trend.test` for alternative tests; `CBData` for constructing a CBData object.

Examples

```
data(shelltox)
if (require(geepack)){
 GEE.trend.test(shelltox, "trend")
}
```

`jointprobs` *Estimate joint event probabilities for multinomial data*

Description

An exchangeable multinomial distribution with $K+1$ categories $O_1, \ldots, O_{K+1}$, can be parameterized by the joint probabilities of events

$$\tau_{r_1,\ldots,r_K|n} = P\left[X_1 = \cdots = X_{r_1} = O_1, \ldots, X_{\sum_{i=1}^{K-1} r_i+1} = \cdots = X_{\sum_{i=1}^{K} r_i} = O_K\right]$$

where $r_i \geq 0$ and $r_1 + \cdots + r_K \leq n$. The `jointprobs` function estimates these probabilities under various settings. Note that when some of the r_i's equal zero, then no restriction on the number of outcomes of the corresponding type are imposed, so the resulting probabilities are marginal.

Usage

```
jointprobs(cmdata, type = c("averaged", "cluster", "mc"))
```

Arguments

`cmdata`	a `CMData` object
`type`	character string describing the desired type of estimate:
	"averaged" - averaged over the observed cluster-size distribution within each treatment
	"cluster" - separately for each cluster size within each treatment
	"mc" - assuming marginal compatibility, ie that τ does not depend on the cluster-size

Value

a list with an array of estimates for each treatment. For a multinomial distribution with $K + 1$ categories the arrays will have either $K + 1$ or K dimensions, depending on whether cluster-size specific estimates (`type="cluster"`) or pooled estimates (`type="averaged"` or `type="mc"`) are requested. For the cluster-size specific estimates the first dimension is the cluster-size. Each additional dimension is a possible outcome.

See Also

`mc.est` for estimating the distribution under marginal compatibility, `uniprobs` and `multi.corr` for extracting the univariate marginal event probabilities, and the within-multinomial correlations from the joint probabilities.

Examples

```
data(dehp)
# averaged over cluster-sizes
tau.ave <- jointprobs(dehp, type="ave")
# averaged P(X1=X2=O1, X3=O2) in the 1500 dose group
tau.ave[["1500"]]["2","1"]  # there are two type-1, and one type-2 outcome

#plot P(X1=O1) - the marginal probability of a type-1 event over cluster-sizes
tau <- jointprobs(dehp, type="cluster")
ests <- as.data.frame(lapply(tau, function(x)x[,"1","0"]))
matplot(ests, type="b")
```

`mc.est.CMData` *Distribution of the number of responses assuming marginal compatibility.*

Description

The `mc.est` function estimates the distribution of the number of responses in a cluster under the assumption of marginal compatibility: information from all cluster sizes is pooled. The estimation is performed independently for each treatment group.

Usage

```
## S3 method for class 'CMData'
mc.est(object, eps = 1e-06, ...)

## S3 method for class 'CBData'
mc.est(object, ...)

mc.est(object, ...)
```

Arguments

object	a CBData or CMData object
eps	numeric; EM iterations proceed until the sum of squared changes fall below eps
...	other potential arguments; not currently used

Details

The EM algorithm given by Stefanescu and Turnbull (2003) is used for the binary data.

Value

For CMData: A data frame giving the estimated pdf for each treatment and clustersize. The probabilities add up to 1 for each Trt/ClusterSize combination. It has the following columns:

Prob	numeric, the probability of NResp responses in a cluster of size ClusterSize in group Trt
Trt	factor, the treatment group
ClusterSize	numeric, the cluster size
NResp.1 - NResp.K	numeric, the number of responses of each type

For CBData: A data frame giving the estimated pdf for each treatment and clustersize. The probabilities add up to 1 for each Trt/ClusterSize combination. It has the following columns:

Prob	numeric, the probability of NResp responses in a cluster of size ClusterSize in group Trt
Trt	factor, the treatment group
ClusterSize	numeric, the cluster size
NResp	numeric, the number of responses

Note

For multinomial data, the implementation is currently written in R, so it is not very fast.

Author(s)

Aniko Szabo

References

George EO, Cheon K, Yuan Y, Szabo A (2016) On Exchangeable Multinomial Distributions. #'*Biometrika* 103(2), 397-408.

Stefanescu, C. & Turnbull, B. W. (2003) Likelihood inference for exchangeable binary data with varying cluster sizes. *Biometrics*, 59, 18-24

Examples

```
data(dehp)
dehp.mc <- mc.est(subset(dehp, Trt=="0"))
subset(dehp.mc, ClusterSize==2)

data(shelltox)
sh.mc <- mc.est(shelltox)

if (require(lattice)){
xyplot(Prob~NResp|factor(ClusterSize), groups=Trt, data=sh.mc, subset=ClusterSize>0,
   type="l", as.table=TRUE, auto.key=list(columns=4, lines=TRUE, points=FALSE),
   xlab="Number of responses", ylab="Probability P(R=r|N=n)")
}
```

`mc.test.chisq.CMData` *Test the assumption of marginal compatibility*

Description

`mc.test.chisq` tests whether the assumption of marginal compatibility is violated in the data.

Usage

```
## S3 method for class 'CMData'
mc.test.chisq(object, ...)

## S3 method for class 'CBData'
mc.test.chisq(object, ...)

mc.test.chisq(object, ...)
```

Arguments

`object`	a `CBData` or `CMData` object
`...`	other potential arguments; not currently used

Details

The assumption of marginal compatibility (AKA reproducibility or interpretability) implies that the marginal probability of response does not depend on clustersize. Stefanescu and Turnbull (2003), and Pang and Kuk (2007) developed a Cochran-Armitage type test for trend in the marginal probability of success as a function of the clustersize. `mc.test.chisq` implements a generalization of that test extending it to multiple treatment groups.

Value

A list with the following components:

`overall.chi`	the test statistic; sum of the statistics for each group
`overall.p`	p-value of the test
`individual`	a list of the results of the test applied to each group separately: • chi.sq the test statistic for the group • p p-value for the group

Author(s)

Aniko Szabo

References

Stefanescu, C. & Turnbull, B. W. (2003) Likelihood inference for exchangeable binary data with varying cluster sizes. *Biometrics*, 59, 18-24

Pang, Z. & Kuk, A. (2007) Test of marginal compatibility and smoothing methods for exchangeable binary data with unequal cluster sizes. *Biometrics*, 63, 218-227

See Also

`mc.est` for estimating the distribution under marginal compatibility.

Examples

```
data(dehp)
mc.test.chisq(dehp)


data(shelltox)
mc.test.chisq(shelltox)
```

---

multi.corr *Extract correlation coefficients from joint probability arrays*

Description

Calculates the within- and between-outcome correlation coefficients for exchangeable correlated multinomial data based on joint probability estimates calculated by the `jointprobs` function. These determine the variance inflation due the cluster structure.

Usage

```
multi.corr(jp, type = attr(jp, "type"))
```

Arguments

jp	the output of jointprobs - a list of joint probability arrays by treatment
type	one of c("averaged","cluster","mc") - the type of joint probability. By default, the type attribute of jp is used.

Details

If R_i and R_j is the number of events of type i and j, respectively, in a cluster of size n, then

$$Var(R_i) = np_i(1-p_i)(1+(n-1)\phi_{ii})$$

$$Cov(R_i, R_j) = -np_ip_j(1+(n-1)\phi_{ij})$$

where p_i and p_j are the marginal event probabilities and ϕ_{ij} are the correlation coefficients computed by multi.corr.

Value

a list of estimated correlation matrices by treatment group. If cluster-size specific estimates were requested ((type="cluster")), then each list elements are a list of these matrices for each cluster size.

See Also

jointprobs for calculating the joint probability arrays

Examples

```
data(dehp)
tau <- jointprobs(dehp, type="averaged")
multi.corr(tau)
```

---

multinom.gen *Functions for generating multinomial outcomes*

Description

These are built-in functions to be used by ran.CMData for generating random multinomial data.

Usage

```
mg.Resample(n, clustersizes, param)

mg.DirMult(n, clustersizes, param)

mg.LogitNorm(n, clustersizes, param)

mg.MixMult(n, clustersizes, param)
```

Arguments

n	number of independent clusters to generate
clustersizes	an integer vector specifying the sizes of the clusters
param	a list of parameters for each specific generator

Details

For **mg.Resample**: the param list should be list(param=...), in which the CMData object to be resampled is passed.

For **mg.DirMult**: the param list should be list(shape=...), in which the parameter vector of the Dirichlet distribution is passed (see rdirichlet).

For **mg.LogitNorm**: the param list should be list(mu=...,sigma=...), in which the mean vector and covariance matrix of the underlying Normal distribution are passed. If sigma is NULL (or missing), then an identity matrix is assumed. They should have *K-1* dimensions for a *K*-variate multinomial.

For **mg.MixMult**: the param list should be list(q=...,m=...), in which the vector of mixture probabilities q and the matrix m of logit-transformed means of each component are passed. For a *K*-variate multinomial, the matrix m should have *K-1* columns and length(q) rows.

NOSTASOT *Finding the NOSTASOT dose*

Description

The NOSTASOT dose is the No-Statistical-Significance-Of-Trend dose – the largest dose at which no trend in the rate of response has been observed. It is often used to determine a safe dosage level for a potentially toxic compound.

Usage

```
NOSTASOT(
  cbdata,
  test = c("RS", "GEE", "GEEtrend", "GEEall", "SO"),
  exact = test == "SO",
  R = 100,
  sig.level = 0.05,
  control = soControl()
)
```

Arguments

cbdata	a CBData object
test	character string defining the desired test statistic. See trend.test for details.
exact	logical, should an exact permutation test be performed. See trend.test for details.

R	integer, number of permutations for the exact test
sig.level	numeric between 0 and 1, significance level of the test
control	an optional list of control settings for the stochastic order ("SO") test, usually a call to soControl. See there for the names of the settable control values and their effect.

Details

A series of hypotheses about the presence of an increasing trend overall, with all but the last group, all but the last two groups, etc. are tested. Since this set of hypotheses forms a closed family, one can test these hypotheses in a step-down manner with the same sig.level type I error rate at each step and still control the family-wise error rate.

The NOSTASOT dose is the largest dose at which the trend is not statistically significant. If the trend test is not significant with all the groups included, the largest dose is the NOSTASOT dose. If the testing sequence goes down all the way to two groups, and a significant trend is still detected, the lowest dose is the NOSTASOT dose. This assumes that the lowest dose is a control group, and this convention might not be meaningful otherwise.

Value

a list with two components

NOSTASOT	character string identifying the NOSTASOT dose.
p	numeric vector of the p-values of the tests actually performed.

The last element corresponds to all doses included, and will not be missing. p-values for tests that were not actually performed due to the procedure stopping are set to NA.

Author(s)

Aniko Szabo, aszabo@mcw.edu

References

Tukey, J. W.; Ciminera, J. L. & Heyse, J. F. (1985) Testing the statistical certainty of a response to increasing doses of a drug. *Biometrics* 41, 295-301.

See Also

trend.test for details about the available trend tests.

Examples

```
data(shelltox)
NOSTASOT(shelltox, test="RS")
```

pdf *Parametric distributions for correlated binary data*

Description

`qpower.pdf` and `betabin.pdf` calculate the probability distribution function for the number of responses in a cluster of the q-power and beta-binomial distributions, respectively.

Usage

```
betabin.pdf(p, rho, n)

qpower.pdf(p, rho, n)
```

Arguments

`p`	numeric, the probability of success.
`rho`	numeric between 0 and 1 inclusive, the within-cluster correlation.
`n`	integer, cluster size.

Details

The pdf of the q-power distribution is

$$P(X = x) = \binom{n}{x} \sum_{k=0}^{x} (-1)^k \binom{x}{k} q^{(n-x+k)^\gamma},$$

$x = 0, \ldots, n$, where $q = 1 - p$, and the intra-cluster correlation

$$\rho = \frac{q^{2^\gamma} - q^2}{q(1-q)}.$$

The pdf of the beta-binomial distribution is

$$P(X = x) = \binom{n}{x} \frac{B(\alpha + x, n + \beta - x)}{B(\alpha, \beta)},$$

$x = 0, \ldots, n$, where $\alpha = p\frac{1-\rho}{\rho}$, and $\alpha = (1-p)\frac{1-\rho}{\rho}$.

Value

a numeric vector of length $n + 1$ giving the value of $P(X = x)$ for $x = 0, \ldots, n$.

Author(s)

Aniko Szabo, aszabo@mcw.edu

References

Kuk, A. A (2004) Litter-based approach to risk assessment in developmental toxicity studies via a power family of completely monotone functions *Applied Statistics*, 52, 51-61.

Williams, D. A. (1975) The Analysis of Binary Responses from Toxicological Experiments Involving Reproduction and Teratogenicity *Biometrics*, 31, 949-952.

See Also

`ran.CBData` for generating an entire dataset using these functions

Examples

```
#the distributions have quite different shapes
#with q-power assigning more weight to the "all affected" event than other distributions
plot(0:10, betabin.pdf(0.3, 0.4, 10), type="o", ylim=c(0,0.34),
  ylab="Density", xlab="Number of responses out of 10")
lines(0:10, qpower.pdf(0.3, 0.4, 10), type="o", col="red")
```

`ran.CBData` *Generate random correlated binary data*

Description

`ran.mc.CBData` generates a random `CBData` object from a given two-parameter distribution.

Usage

```
ran.CBData(
  sample.sizes,
  p.gen.fun = function(g) 0.3,
  rho.gen.fun = function(g) 0.2,
  pdf.fun = qpower.pdf
)
```

Arguments

`sample.sizes`	a dataset with variables Trt, ClusterSize and Freq giving the number of clusters to be generated for each Trt/ClusterSize combination.
`p.gen.fun`	a function of one parameter that generates the value of the first parameter of `pdf.fun` (*p*) given the group number.
`rho.gen.fun`	a function of one parameter that generates the value of the second parameter of `pdf.fun` (*rho*) given the group number.
`pdf.fun`	a function of three parameters (*p, rho, n*) giving the PDF of the number of responses in a cluster given the two parameters (*p, rho*), and the cluster size (*n*). Functions implementing two common distributions: the beta-binomial (`betabin.pdf`) and q-power (`qpower.pdf`) are provided in the package.

Details

p.gen.fun and *rho.gen.fun* are functions that generate the parameter values for each treatment group; *pdf.fun* is a function generating the pdf of the number of responses given the two parameters *p* and *rho*, and the cluster size *n*.

`p.gen.fun` and `rho.gen.fun` expect the parameter value of 1 to represent the first group, 2 - the second group, etc.

Value

a CBData object with randomly generated number of responses with sample sizes specified in the call.

Author(s)

Aniko Szabo, aszabo@mcw.edu

See Also

`betabin.pdf` and `qpower.pdf`

Examples

```
set.seed(3486)
ss <- expand.grid(Trt=0:3, ClusterSize=5, Freq=4)
#Trt is converted to a factor
rd <- ran.CBData(ss, p.gen.fun=function(g)0.2+0.1*g)
rd
```

---

ran.CMData *Generate a random CMData object*

Description

Generates random exchangeably correlated multinomial data based on a parametric distribution or using resampling. The Dirichlet-Multinomial, Logistic-Normal multinomial, and discrete mixture multinomial parametric distributions are implemented. All observations will be assigned to the same treatment group, and there is no guarantee of a specific order of the observations in the output.

Usage

```
ran.CMData(n, ncat, clustersize.gen, distribution)
```

Arguments

n	number of independent clusters to generate
ncat	number of response categories
clustersize.gen	either an integer vector specifying the sizes of the clusters, which will be recycled to achieve the target number of clusters n; or a function with one parameter that returns an integer vector of cluster sizes when the target number of clusters n is passed to it as a parameter
distribution	a list with two components: "multinom.gen" and "param" that specifies the generation process for each cluster. The "multinom.gen" component should be a function of three parameters: number of clusters, vector of cluster sizes, and parameter list, that a matrix of response counts where each row is a cluster and each column is the number of responses of a given type. The "param" component should specify the list of parameters needed by the multinom.gen function.

Value

a CMData object with randomly generated number of responses with sample sizes specified in the call

Author(s)

Aniko Szabo

See Also

CMData for details about CMData objects; multinom.gen for built-in generating functions

Examples

```
# Resample from the dehp dataset
data(dehp)
ran.dehp <- ran.CMData(20, 3, 10, list(multinom.gen=mg.Resample, param=list(data=dehp)))

# Dirichlet-Multinomial distribution with two treatment groups and random cluster sizes
binom.cs <- function(n){rbinom(n, p=0.3, size=10)+1}
dm1 <- ran.CMData(15, 4, binom.cs,
                  list(multinom.gen=mg.DirMult, param=list(shape=c(2,3,2,1))))
dm2 <- ran.CMData(15, 4, binom.cs,
                  list(multinom.gen=mg.DirMult, param=list(shape=c(1,1,4,1))))
ran.dm <- rbind(dm1, dm2)

# Logit-Normal multinomial distribution
ran.ln <- ran.CMData(13, 3, 3,
                     list(multinom.gen=mg.LogitNorm,
                          param=list(mu=c(-1, 1), sigma=matrix(c(1,0.8,0.8,2), nr=2))))

# Mixture of two multinomial distributions
unif.cs <- function(n){sample(5:9, size=n, replace=TRUE)}
ran.mm <- ran.CMData(6, 3, unif.cs,
```

```
list(multinom.gen=mg.MixMult,
     param=list(q=c(0.8,0.2), m=rbind(c(-1,0), c(0,2)))))
```

`read.CBData` *Read data from external file into a CBData object*

Description

A convenience function to read data from specially structured file directly into a `CBData` object.

Usage

```
read.CBData(file, with.freq = TRUE, ...)
```

Arguments

`file`	name of file with data. The first column should contain the treatment group, the second the size of the cluster, the third the number of responses in the cluster. Optionally, a fourth column could give the number of times the given combination occurs in the data.
`with.freq`	logical indicator of whether a frequency variable is present in the file
`...`	additional arguments passed to `read.table`

Value

a `CBData` object

Author(s)

Aniko Szabo

See Also

`CBData`

`read.CMData` *Read data from external file into a CMData object*

Description

A convenience function to read data from specially structured file directly into a `CMData` object. There are two basic data format options: either the counts of responses of all categories are given (and the cluster size is the sum of these counts), or the total cluster size is given with the counts of all but one category. The first column should always give the treatment group, then either the counts for each category (first option, chosen by setting `with.clustersize = FALSE`), or the size of the cluster followed by the counts for all but one category (second option, chosen by setting `with.clustersize = TRUE`). Optionally, a last column could give the number of times the given combination occurs in the data.

Usage

```
read.CMData(file, with.clustersize = TRUE, with.freq = TRUE, ...)
```

Arguments

`file`	name of file with data. The data in the file should be structured as described above.
`with.clustersize`	logical indicator of whether a cluster size variable is present in the file
`with.freq`	logical indicator of whether a frequency variable is present in the file
`...`	additional arguments passed to `read.table`

Value

a `CMData` object

Author(s)

Aniko Szabo

See Also

`CMData`

RS.trend.test *Rao-Scott trend test*

Description

RS.trend.test implements the Rao-Scott adjusted Cochran-Armitage test for linear increasing trend with correlated data.

Usage

```
RS.trend.test(cbdata)
```

Arguments

cbdata	a CBData object

Details

The test is based on calculating a *design effect* for each cluster by dividing the observed variability by the one expected under independence. The number of responses and the cluster size are then divided by the design effect, and a Cochran-Armitage type test statistic is computed based on these adjusted values.

The implementation aims for testing for *increasing* trend, and a one-sided p-value is reported. The test statistic is asymptotically normally distributed, and a two-sided p-value can be easily computed if needed.

Value

A list with components

statistic	numeric, the value of the test statistic
p.val	numeric, asymptotic one-sided p-value of the test

Author(s)

Aniko Szabo, aszabo@mcw.edu

References

Rao, J. N. K. & Scott, A. J. A (1992) Simple Method for the Analysis of Clustered Data *Biometrics*, 48, 577-586.

See Also

SO.trend.test, GEE.trend.test for alternative tests; CBData for constructing a CBData object.

Examples

```
data(shelltox)
RS.trend.test(shelltox)
```

---

shelltox *Shell Toxicology data*

Description

This is a classical developmental toxicology data set. Pregnant banded Dutch rabbits were treated with one of four levels of a chemical. The actual doses are not known, instead the groups are designated as Control, Low, Medium, and High. Before term the animals were sacrificed, and the total number of fetuses, as well as the number affected by the treatment was recorded.

Usage

```
data(shelltox)
```

Format

A 'CBData' object, that is a data frame with the following variables

Trt	factor giving treatment group
ClusterSize	the size of the litter
NResp	the number of affected fetuses
Freq	the number of litters with the given ClusterSize/NResp combination

Source

Paul, S. R. (1982) Analysis of proportions of affected foetuses in teratological experiments. *Biometrics*, 38, 361-370.

This data set has been analyzed (and listed) in numerous papers, including

Rao, J. N. K. & Scott, A. J. (1992) A Simple Method for the Analysis of Clustered Data. *Biometrics*, 48, 577-586.

George, E. O. & Kodell, R. L. (1996) Tests of Independence, Treatment Heterogeneity, and Dose-Related Trend With Exchangeable Binary Data. *Journal of the American Statistical Association*, 91, 1602-1610.

Lee, S. (2003) Analysis of the Binary Littermate Data in the One-Way Layout. *Biometrical Journal*, 45, 195-206.

Examples

```
data(shelltox)
stripchart(I(NResp/ClusterSize)~Trt, cex=sqrt(shelltox$Freq), data=shelltox, pch=1,
           method="jitter", vertical=TRUE, ylab="Proportion affected")
```

SO.LRT *Likelihood-ratio test statistic*

Description

SO.LRT computes the likelihood ratio test statistic for stochastic ordering against equality assuming marginal compatibility for both alternatives. Note that this statistic does not have a χ^2 distribution, so the p-value computation is not straightforward. The SO.trend.test function implements a permutation-based evaluation of the p-value for the likelihood-ratio test.

Usage

```
SO.LRT(cbdata, control = soControl())
```

Arguments

cbdata	a CBData object
control	an optional list of control settings, usually a call to soControl. See there for the names of the settable control values and their effect.

Value

The value of the likelihood ratio test statistic is returned with two attributes:

ll0	the log-likelihood under H_0 (equality)
ll1	the log-likelihood under H_a (stochastic order)

Author(s)

Aniko Szabo

See Also

SO.trend.test, soControl

Examples

```
data(shelltox)
LRT <- SO.LRT(shelltox, control=soControl(max.iter = 100, max.directions = 50))
LRT
```

SO.mc.est *Order-restricted MLE assuming marginal compatibility*

Description

SO.mc.est computes the nonparametric maximum likelihood estimate of the distribution of the number of responses in a cluster $P(R = r|n)$ under a stochastic ordering constraint. Umbrella ordering can be specified using the turn parameter.

Usage

```
SO.mc.est(cbdata, turn = 1, control = soControl())
```

Arguments

cbdata	an object of class CBData.
turn	integer specifying the peak of the umbrella ordering (see Details). The default corresponds to a non-decreasing order.
control	an optional list of control settings, usually a call to soControl. See there for the names of the settable control values and their effect.

Details

Two different algorithms: EM and ISDM are implemented. In general, ISDM (the default) should be faster, though its performance depends on the tuning parameter max.directions: values that are too low or too high slow the algorithm down.

SO.mc.est allows extension to an umbrella ordering: $D_1 \geq^{st} \cdots \geq^{st} D_k \leq^{st} \cdots \leq^{st} D_n$ by specifying the value of k as the turn parameter. This is an experimental feature, and at this point none of the other functions can handle umbrella orderings.

Value

A list with components:

Components Q and D are unlikely to be needed by the user.

MLest	data frame with the maximum likelihood estimates of $P(R_i = r|n)$
Q	numeric matrix; estimated weights for the mixing distribution
D	numeric matrix; directional derivative of the log-likelihood
loglik	the achieved value of the log-likelihood
converge	a 2-element vector with the achieved relative error and the performed number of iterations

Author(s)

Aniko Szabo, aszabo@mcw.edu

References

Szabo A, George EO. (2010) On the Use of Stochastic Ordering to Test for Trend with Clustered Binary Data. *Biometrika* 97(1), 95-108.

See Also

soControl

Examples

```
data(shelltox)
ml <- SO.mc.est(shelltox, control=soControl(eps=0.01, method="ISDM"))
attr(ml, "converge")

require(lattice)
panel.cumsum <- function(x,y,...){
  x.ord <- order(x)
  panel.xyplot(x[x.ord], cumsum(y[x.ord]), ...)}

xyplot(Prob~NResp|factor(ClusterSize), groups=Trt, data=ml, type="s",
     panel=panel.superpose, panel.groups=panel.cumsum,
     as.table=TRUE, auto.key=list(columns=4, lines=TRUE, points=FALSE),
     xlab="Number of responses", ylab="Cumulative Probability R(R>=r|N=n)",
    ylim=c(0,1.1), main="Stochastically ordered estimates\n with marginal compatibility")
```

SO.trend.test *Likelihood ratio test of stochastic ordering*

Description

Performs a likelihood ratio test of stochastic ordering versus equality using permutations to estimate the null-distribution and the p-value. If only the value of the test statistic is needed, use SO.LRT instead.

Usage

```
SO.trend.test(cbdata, R = 100, control = soControl())
```

Arguments

cbdata	a CBData object.
R	an integer – the number of random permutations for estimating the null distribution.
control	an optional list of control settings, usually a call to soControl. See there for the names of the settable control values and their effect.

Details

The test is valid only under the assumption that the cluster-size distribution does not depend on group. During the estimation of the null-distribution the group assignments of the clusters are permuted keeping the group sizes constant; the within-group distribution of the cluster-sizes will vary randomly during the permutation test.

The default value of R is probably too low for the final data analysis, and should be increased.

Value

A list with the following components

LRT	the value of the likelihood ratio test statistic. It has two attributes: ll0 and ll1 - the values of the log-likelihood under H_0 and H_a respectively.
p.val	the estimated one-sided p-value.
boot.res	an object of class "boot" with the detailed results of the permutations. See boot for details.

Author(s)

Aniko Szabo, aszabo@mcw.edu

References

Szabo A, George EO. (2010) On the Use of Stochastic Ordering to Test for Trend with Clustered Binary Data. *Biometrika* 97(1), 95-108.

See Also

SO.LRT for calculating only the test statistic, soControl

Examples

```
data(shelltox)
set.seed(45742)
sh.test <- SO.trend.test(shelltox, R=10, control=soControl(eps=0.1, max.directions=25))
sh.test

#a plot of the resampled LRT values
#would look better with a reasonable value of R
null.vals <- sh.test$boot.res$t[,1]
hist(null.vals, breaks=10,  freq=FALSE, xlab="Test statistic", ylab="Density",
     main="Simulated null-distribution", xlim=range(c(0,20,null.vals)))
points(sh.test$LRT, 0, pch="*",col="red", cex=3)
```

soControl *Control values for order-constrained fit*

Description

The values supplied in the function call replace the defaults and a list with all possible arguments is returned. The returned list is used as the control argument to the mc.est, SO.LRT, and SO.trend.test functions.

Usage

```
soControl(
  method = c("ISDM", "EM"),
  eps = 0.005,
  max.iter = 5000,
  max.directions = 0,
  start = ifelse(method == "ISDM", "H0", "uniform"),
  verbose = FALSE
)
```

Arguments

method	a string specifying the maximization method
eps	a numeric value giving the maximum absolute error in the log-likelihood
max.iter	an integer specifying the maximal number of iterations
max.directions	an integer giving the maximal number of directions considered at one step of the ISDM method. If zero or negative, it is set to the number of non-empty cells. A value of 1 corresponds to the VDM algorithm.
start	a string specifying the starting setup of the mixing distribution; "H0" puts weight only on constant vectors (corresponding to the null hypothesis of no change), "uniform" puts equal weight on all elements. Only a "uniform" start can be used for the "EM" algorithm.
verbose	a logical value; if TRUE details of the optimization are shown.

Value

a list with components for each of the possible arguments.

Author(s)

Aniko Szabo aszabo@mcw.edu

See Also

mc.est, SO.LRT, SO.trend.test

Examples

```
# decrease the maximum number iterations and
# request the "EM" algorithm
soControl(method="EM", max.iter=100)
```

`trend.test` *Test for increasing trend with correlated binary data*

Description

The `trend.test` function provides a common interface to the trend tests implemented in this package: `SO.trend.test`, `RS.trend.test`, and `GEE.trend.test`. The details of each test can be found on their help page.

Usage

```
trend.test(
  cbdata,
  test = c("RS", "GEE", "GEEtrend", "GEEall", "SO"),
  exact = test == "SO",
  R = 100,
  control = soControl()
)
```

Arguments

`cbdata`	a `CBData` object
`test`	character string defining the desired test statistic. "RS" performs the Rao-Scott test (`RS.trend.test`), "SO" performs the stochastic ordering test (`SO.trend.test`), "GEE", "GEEtrend", "GEEall" perform the GEE-based test (`GEE.trend.test`) with constant, linearly modeled, and freely varying scale parameters, respectively.
`exact`	logical, should an exact permutation test be performed. Only an exact test can be performed for "SO". The default is to use the asymptotic p-values except for "SO".
`R`	integer, number of permutations for the exact test
`control`	an optional list of control settings for the stochastic order ("SO") test, usually a call to `soControl`. See there for the names of the settable control values and their effect.

Value

A list with two components and an optional "boot" attribute that contains the detailed results of the permutation test as an object of class `boot` if an exact test was performed.

`statistic`	numeric, the value of the test statistic
`p.val`	numeric, asymptotic one-sided p-value of the test

Author(s)

Aniko Szabo, aszabo@mcw.edu

See Also

SO.trend.test, RS.trend.test, and GEE.trend.test for details about the available tests.

Examples

```
data(shelltox)
trend.test(shelltox, test="RS")
set.seed(5724)
#R=50 is too low to get a good estimate of the p-value
trend.test(shelltox, test="RS", R=50, exact=TRUE)
```

---

uniprobs *Extract univariate marginal probabilities from joint probability arrays*

Description

Calculates the marginal probability of each event type for exchangeable correlated multinomial data based on joint probability estimates calculated by the jointprobs function.

Usage

```
uniprobs(jp, type = attr(jp, "type"))
```

Arguments

jp	the output of jointprobs - a list of joint probability arrays by treatment
type	one of c("averaged","cluster","mc") - the type of joint probability. By default, the type attribute of jp is used.

Value

a list of estimated probability of each outcome by treatment group. The elements are either matrices or vectors depending on whether cluster-size specific estimates were requested (type="cluster") or not.

See Also

jointprobs for calculating the joint probability arrays

Examples

```
data(dehp)
tau <- jointprobs(dehp, type="averaged")
uniprobs(tau)

#separately for each cluster size
tau2 <- jointprobs(dehp, type="cluster")
uniprobs(tau2)
```

`unwrap.CBData` *Unwrap a clustered object*

Description

`unwrap` is a utility function that reformats a CBData or CMData object so that each row is one observation (instead of one or more clusters). A new 'ID' variable is added to indicate clusters. This form can be useful for setting up the data for a different package.

Usage

```
## S3 method for class 'CBData'
unwrap(object, ...)

## S3 method for class 'CMData'
unwrap(object, ...)

unwrap(object, ...)
```

Arguments

`object`	a `CBData` object
`...`	other potential arguments; not currently used

Value

For `unwrap.CMData`: a data frame with one row for each cluster element (having a multinomial outcome) with the following standardized column names

`Trt`	factor, the treatment group
`ClusterSize`	numeric, the cluster size
`ID`	factor, each level representing a different cluster
`Resp`	numeric with integer values giving the response type of the cluster element

For `unwrap.CBData`: a data frame with one row for each cluster element (having a binary outcome) with the following standardized column names

`Trt`	factor, the treatment group

ClusterSize	numeric, the cluster size
ID	factor, each level representing a different cluster
Resp	numeric with 0/1 values, giving the response of the cluster element

Author(s)

Aniko Szabo

Examples

```
data(dehp)
dehp.long <- unwrap(dehp)
head(dehp.long)


data(shelltox)
ush <- unwrap(shelltox)
head(ush)
```

Index