Package 'hmmm'

October 13, 2022

Version 1.0-4

Date 2018-03-05

Title Hierarchical Multinomial Marginal Models

Author Colombi Roberto and Sabrina Giordano and Manuela Cazzaro, with contributions from Joseph Lang

Maintainer Colombi Roberto <colombi@unibg.it>

Description Functions for specifying and fitting marginal models for contingency tables proposed by Bergsma and Rudas (2002) here called hierarchical multinomial marginal models (hmmm) and their extensions presented by Bartolucci et al.
(2007); multinomial Poisson homogeneous (mph) models and homogeneous linear predictor (hlp) models for contingency
tables proposed by Lang (2004) and (2005); hidden Markov models where the distribution of the observed variables
is described by a marginal model.
Inequality constraints on the parameters are allowed and can be tested.

Imports quadprog, MASS, mvtnorm, nleqslv

License GPL (>= 2)

Repository CRAN

URL https://www.r-project.org

NeedsCompilation no

Date/Publication 2018-03-14 11:26:58 UTC

R topics documented:

hmmm-package 2
accident 3
akaike 4
anova.hidden 5
anova.hmmmfit 6
create.XMAT 7
depression 9
drinks 10

getnames . . . 10
GMI . . . 11
hidden.emfit . . . 12
hmmm.chibar . . . 14
hmmm.mlfit . . . 16
hmmm.model . . . 18
hmmm.model.X . . . 21
inv_GMI . . . 22
kentucky . . . 23
loglin.model . . . 24
madsen . . . 25
marg.list . . . 26
mphineq.fit . . . 27
polbirth . . . 30
print.hidden . . . 31
print.hmmmchibar . . . 31
print.hmmmfit . . . 32
recursive . . . 33
relpol . . . 35
relpolbirth . . . 36
summary.hidden . . . 36
summary.hmmmchibar . . . 37
summary.hmmmfit . . . 38
summary.hmmmmod . . . 39
summary.mphfit . . . 40

Index **42**

`hmmm-package` *package hmmm*

Description

Functions for specifying and fitting marginal models for contingency tables proposed by Bergsma and Rudas (2002) here called hierarchical multinomial marginal models (hmmm) and their extensions presented by Bartolucci et al (2007); multinomial Poisson homogeneous (mph) models and homogeneous linear predictor (hlp) models for contingency tables proposed by Lang (2004, 2005); hidden Markov models where the distribution of the observed variables is described by a marginal model. Inequality constraints on the parameters are allowed and can be tested.

Author(s)

Roberto Colombi, Sabrina Giordano and Manuela Cazzaro. Joseph B. Lang is the author of functions 'num.deriv.fct', 'create.U' for mph models.

References

Bergsma WP, Rudas T (2002) Marginal models for categorical data. The Annals of Statistics, 30, 140-159

Bartolucci F, Colombi R, Forcina A (2007) An extended class of marginal link functions for modelling contingency tables by equality and inequality constraints. Statistica Sinica, 17, 691-711

Lang JB (2004) Multinomial Poisson homogeneous models for contingency tables. The Annals of Statistics, 32, 340-383

Lang JB (2005) Homogeneous linear predictor models for contingency tables. Journal of the American Statistical Association, 100, 121-134.

`accident` *factory accident data*

Description

Data on factory accidents occurred in Bergamo (Italy) in 1998, collected by the Inail (Italian institute for insurance against factory accidents): 1052 workers who suffered an accident and claimed for a compensation are classified according to the type of injury, the time to recover (number of working days lost), the age (years), and the solar hour (part of the day in which the accident occurred).

Usage

```
data(accident)
```

Format

A data frame whose columns contain:

`Type` A factor with levels: `uncertain`, `avoidable`, `not-avoidable`

`Time` A factor with levels: `0 |-- 7`, `7 |-- 21`, `21 |-- 60`, `>= 60`

`Age` A factor with levels: `<= 25`, `26 -- 45`, `> 45`

`Hour` A factor with levels: `morning`, `afternoon`

`Freq` A numeric vector of frequencies

Source

Inail, Bergamo (Italy) 1998

References

Cazzaro M, Colombi R (2008) Modelling two way contingency tables with recursive logits and odds ratios. Statistical Methods and Applications, 17, 435-453.

Examples

```
data(accident)
```

akaike *akaike criterium*

Description

Compute AIC value for a list of hmm and hidden Markov models

Usage

```
akaike(..., LRTEST = FALSE, ORDERED = FALSE, NAMES = NULL)
```

Arguments

...	objects created by 'hmmm.mlfit' or 'hidden.emfit'
LRTEST	If TRUE, the first model must include all the others models as special cases. For every model, the likelihood ratio statistic test with respect to the first model is computed
ORDERED	If TRUE, in the output the models are ordered according to the Akaike criterium starting from the lowest AIC value
NAMES	Optional character vector with the names of the models. If it is NULL (the default) model names are created as model1, model2.....

Details

The models in input must be at least two objects of the classes `hmmmfit` or `hidden`.

Value

A matrix with row names given by `NAMES` and column names describing the output for every model (position of the model in the input list `#model`, the loglikelihood function `loglik`, the number of parameters `npar`, the number of constraints of the model `dfmodel`, likelihood ratio test `LRTEST`, degrees of freedom `dftest`, `PVALUE`, `AIC`, `DELTAAIC`). The `DELTAAIC` is the difference between the AIC value of every model and the lowest AIC value.

References

Konishi S, Kitagawa G (2008) Information criteria and statistical modeling. Springer.

Examples

```
data(madsen)
# 1 = Influence; 2 = Satisfaction;  3 = Contact; 4 = Housing
names<-c("Inf","Sat","Co","Ho")
y<-getnames(madsen,st=6)

margin <- marg.list(c("marg-marg-l-l", "g-marg-l-l", "marg-g-l-l", "g-g-l-l"))

# additive effect of 3 and 4 on logits of 1 in marginal
```

```
# distribution {1, 3, 4}, conditional independence 2_||_3|4
modelA <- hmmm.model(marg = margin, lev = c(3, 3, 2, 4), names = names)
modA <- hmmm.mlfit(y, modelA)
modA

# additive effect of 3 and 4 on logits of 1 in marginal
# distributions {1, 3, 4} and {2, 3, 4}
modelB <- hmmm.model(marg = margin, lev = c(3, 3, 2, 4),
names = names, sel = c(18:23, 34:39))
modB <- hmmm.mlfit(y, modelB)
modB

# 1 and 2 do not depend on the levels of 3 and 4
modelC <- hmmm.model(marg = margin, lev = c(3, 3, 2, 4),
names = names, sel = c(18:23, 34:39, 44:71))
modC <- hmmm.mlfit(y, modelC)
modC

akaike(modB, modA, modC, ORDERED = TRUE, NAMES = c("modB", "modA", "modC"))
akaike(modB, modA, modC, LRTEST = TRUE, NAMES = c("modB", "modA", "modC"))
```

`anova.hidden` *anova for the class hidden*

Description

The generic function 'anova' is adapted to the objects inheriting from class `hidden` (anova.hidden) to compute the likelihood ratio test for nested hidden models estimated by 'hidden.emfit'.

Usage

```
## S3 method for class 'hidden'
anova(object,objectlarge,...)
```

Arguments

`object`	Object of the class `hidden`, reduced model, i.e. modelA
`objectlarge`	Object of the class `hidden`, large model, i.e. modelB
`...`	Other models and further arguments passed to or from other methods

Details

Nested models, fitted by 'hidden.emfit', are compared (e.g. modelA is nested in modelB), the likelihood ratio statistic with the degrees of freedom and the associated pvalue is returned.

See Also

`hidden.emfit`, `summary.hidden`, `print.hidden`

Examples

```
data(drinks)
y<-cbind(drinks$lemon.tea,drinks$orange.juice)

f<-~lat*tea+lat*juice+tea*juice  # lat indicates the latent variable
fm<-c("l-l-l")
fmargobs<-marg.list(fm,mflag="m")
Ptr<-matrix(c(0.941, 0.199,
              0.059, 0.801),2,2,byrow=TRUE)
Ptobs<-matrix(c(0.053, 0.215, 0.206, 0.001, 0.039, 0.021, 0.020, 0.176, 0.270,
                0.000, 0.000, 0.000, 0.048, 0.263, 0.360, 0.065, 0.053, 0.211)
                ,2,9,byrow=TRUE)

# saturated model (fsat<-~lat*tea*juice is implicit)
model.obsf<-hmmm.model(marg=fmargobs,
lev=c(2,3,3),names=c("lat","tea","juice"))
modelsat<-hidden.emfit(y,model.obsf,y.eps=0.01,maxit=10,
maxiter=2500,norm.diff.conv=0.001,old.tran.p=Ptr,bb=Ptobs)

# model with constant association
model.coass<-hmmm.model(marg=fmargobs,
lev=c(2,3,3),names=c("lat","tea","juice"),formula=f)
modelca<-hidden.emfit(y,model.coass,y.eps=0.01,maxit=10,
maxiter=2500,norm.diff.conv=0.001,old.tran.p=Ptr,bb=Ptobs)

a<-anova(modelca,modelsat)
```

`anova.hmmmfit` *anova for the class hmmmfit*

Description

The generic function 'anova' is adapted to the objects inheriting from class `hmmmfit` (anova.hmmmfit) to compute the likelihood ratio test for nested hmm models estimated by 'hmmm.mlfit'.

Usage

```
## S3 method for class 'hmmmfit'
anova(object,objectlarge,...)
```

Arguments

`object`	Object of the class `hmmmfit`, reduced model, i.e. modelA
`objectlarge`	Object of the class `hmmmfit`, large model, i.e. modelB
`...`	Other models and further arguments passed to or from other methods

Details

Nested models, fitted by 'hmmm.mlfit', are compared (e.g. modelA is nested in modelB), the likelihood ratio statistic with the degrees of freedom and the associated pvalue is returned.

See Also

hmmm.mlfit, summary.hmmmfit, print.hmmmfit

Examples

```
data(madsen)
y<-getnames(madsen)
names<-c("Infl","Sat","Co","Ho")

fA<-~Co*Ho+Sat*Co+Sat*Ho
modelA<-loglin.model(lev=c(3,3,2,4),formula=fA,names=names)
fB<-~Co*Ho+Sat*Co+Infl*Co+Sat*Ho+Infl*Sat
modelB<-loglin.model(lev=c(3,3,2,4),formula=fB,names=names)

modA<-hmmm.mlfit(y,modelA)
modB<-hmmm.mlfit(y,modelB)

anova(modA,modB)
```

---

create.XMAT *design matrix for a hmm model*

Description

Function to specify the matrix X of the linear predictor Cln(Mm)=Xbeta for a hmm model.

Usage

```
create.XMAT(model, Formula = NULL,
strata = 1, fnames = NULL, cocacontr = NULL,
ncocacontr = NULL, replace = TRUE)
```

Arguments

model	Object created by 'hmmm.model'
Formula	List of model-formulas; one formula for every marginal interaction
strata	Number of categories of the factors that describe the strata
fnames	Names of the factors that describe the strata
cocacontr	A list of zero-one matrices to build "r" logits created by the function 'recursive'
ncocacontr	Number of contrasts for every factor, if NULL the maximum number is used
replace	If TRUE a new model object with design matrix X is produced, if FALSE the list of design matrices associated to each element specified in Formula is returned

Details

When the marginal interactions of a hmm model are defined in terms of a linear predictor of covariates Cln(Mm)=Xbeta, the list of model formulas defines additive effects of covariates on the interactions. In a case with two response variables declared by `names<-c("A","B")` and two covariates, named C and D by `fnames=c("C","D")`, the additive effect of the covariates on marginal logits of A and B and log odds ratios (A.B) of the two responses is specified by the following `Formula`: `Formula<-list(A=~A*(C+D), B=~B*(C+D), A.B=~A.B*(C+D))`. Use "zero" to constrain to zero all the interactions of a given type.

Value

A list of matrices or a hmm model with X as design matrix according to the input argument `replace`. The parameters beta in the predictor Cln(Mm)=Xbeta are the effects specified in `Formula` and correspond to the columns of X.

References

Lang JB (2004) Multinomial Poisson homogeneous models for contingency tables. The Annals of Statistics, 32, 340-383.

Lang JB (2005) Homogeneous linear predictor models for contingency tables. Journal of the American Statistical Association, 100, 121-134.

See Also

`hmmm.model`, `hmmm.mlfit`, `summary.hmmmfit`

Examples

```
data(accident)
y<-getnames(accident,st=9,sep=";")
# responses: 1 = Type, 2 = Time; covariates: 3 = Age, 4 = Hour

marglist<-c("l-m","m-g","l-g")
marginals<-marg.list(marglist,mflag="m")
names<-c("Type","Time")

modelsat<-hmmm.model(marg=marginals,lev=c(3,4),
strata=6, names=names)

# Create X to account for additive effect of Age and Hour on the logits of Type and Time
# and constant association between Type and Time
al<-list(Type=~Type*(Age+Hour),
Time=~Time*(Age+Hour),Type.Time=~Type.Time)
# list of matrices (replace=FALSE)
listmat<-create.XMAT(modelsat,Formula=al,strata=c(3,2),fnames=c("Age","Hour"),replace=FALSE)

# the model obtained by the modified X (replace=TRUE)
model<-create.XMAT(modelsat,Formula=al,strata=c(3,2),fnames=c("Age","Hour"))
fitmodel<-hmmm.mlfit(y,model,y.eps=0.00001,maxit=2000)
print(fitmodel)
```

depression *longitudinal study of mental depression*

Description

A longitudinal study comparing a new drug with a standard drug for treatment of 340 subjects suffering mental depression (Koch et al., 1977, Agresti, 2013). The patients are classified according to the severity of the initial diagnosis (mild and severe), the treatment type they received (standard and new drugs) and the responses on the depression assessment (normal and abnormal) at three occasions: after one week (R1), two (R2), and four weeks (R3) of treatment.

Usage

```
data(depression)
```

Format

A data frame whose columns contain:

`R3` A factor with levels: `N` as normal, `A` as abnormal

`R2` A factor with levels: `N` as normal, `A` as abnormal

`R1` A factor with levels: `N` as normal, `A` as abnormal

`Treatment` A factor with levels: `standard`, `new drug`

`Diagnosis` A factor with levels: `mild`, `severe`

`Freq` A numeric vector of frequencies

References

Agresti A (2013) Categorical Data Analysis (third edition). Wiley.

Koch GG, Landis JR, Freeman JL, Freeman DH and Lehnen RG (1977) A general metodology for the analysis of experiments with repeated measurement of categorical data. Biometrics, 38, 563-595.

Examples

```
data(depression)
```

drinks *soft-drinks data*

Description

A one-year categorical time series of daily sales of soft-drinks.

Usage

```
data(drinks)
```

Format

A five-variate time series of sale levels of soft-drinks on 269 days

`lemon.tea` A factor with levels: 1 = low, 2 = medium, 3 = high

`orange.juice` A factor with levels: 1 = low, 2 = medium, 3 = high

`apple.juice` A factor with levels: 1 = low, 2 = medium, 3 = high

`mint.tea` A factor with levels: 1 = low, 2 = high

`soya.milk` A factor with levels: 1 = low, 2 = high

References

Colombi R, Giordano S (2011) Lumpability for discrete hidden Markov models. Advances in Statistical Analysis, 95(3), 293-311.

Examples

```
data(drinks)
```

getnames *vector of frequencies from a data frame*

Description

Function to extract the vector of frequencies associated with the category names from a data frame.

Usage

```
getnames(dat, st = 3, sep = " ")
```

Arguments

`dat`	A contingency table in data frame format
`st`	Length of the string for every category name
`sep`	Separator of category names

Value

The function returns a column vector of the frequencies of every combination of categories of the involved variables in the data frame `dat` where each column corresponds to a variable, each row to a combination of categories and the last column reports the frequencies. The variables are arranged so that the farther to the left the column is the faster the category changes. Every frequency of each combination of categories is associated with a string of short category names. The length of the names is determined by setting `st` and consecutive names are separated by the symbol declared by `sep`.

Examples

```
data(madsen)
y<-getnames(madsen,st=3,sep=";")
```

GMI *function to compute the generalized marginal interactions associated to a hierarchical family of marginal sets*

Description

Given a vector of joint probabilities, the generalized marginal interactions (gmi) associated to a hierarchical family of marginal sets are computed. If the input is a matrix, gmi are computed for every column.

Usage

```
GMI(freq, marg, lev, names, mflag = "M")
```

Arguments

`freq`	Matrix of joint probabilities. Every column describes a joint pdf.
`marg`	A character vector decribing the marginal sets and the logits used to build the interactions. See `marg.list`
`lev`	Number of categories of the categorical variables. See the help of `hmmm.model`
`names`	Names of the categorical variables
`mflag`	The symbol used to denote variables that are marginalized, default "M". See `marg.list`

Value

A list with two components: marginals and gmi; marginals is a legend that explains the interactions, gmi is a vector or a matrix that contains the interactions.

References

Colombi R, Giordano S, Cazzaro M (2014) hmmm: An R Package for hierarchical multinomial marginal models. Journal of Statistical Software, 59(11), 1-25, URL http://www.jstatsoft.org/v59/i11/.

See Also

inv_GMI, hmmm.model, marg.list

Examples

```
# joint frequencies for two ordinal variables
# H: level of happiness on a scale from 1 to 5
# S: level of satisfaction on a scale from 1 to 5

y<-c(50,36,15,15,13,15,84,60,42,
     35,6,26,105,113,57,5,26,62,
     465,334,4,10,34,186,1404)

lev<-c(5,5)
marg<-c("g-m","m-g","g-g")
names<-c("H","S")

o<-GMI(cbind(c(y),c(y/sum(y))),marg,lev,names,mflag="m")
o
```

---

hidden.emfit *ML estimation of a multinomial hidden Markov model*

Description

Maximum likelihood estimation of a hidden Markov model with several categorical observed and latent variables. Observed and latent processes are specified by hmm models.

Usage

```
hidden.emfit(y, model.obs, model.lat, nlat = 1, noineq = TRUE, maxit = 10,
maxiter = 100,
norm.diff.conv = 1e-05, norm.score.conv = 1e-05, y.eps = 0, mup = 1, step = 1,
printflag = 0, old.tran.p = NULL, bb = NULL, q.par=1)
```

Arguments

y	The observed multivariate categorical time series
model.obs	The model for the observations specified by 'hmmm.model' or 'hmmm.model.X'
model.lat	The model for the latent chain specified by 'hmmm.model' or 'hmmm.model.X'
nlat	The number of latent variables
noineq	If TRUE inequality constraints are not used
maxit	Maximum number of iterations for the M step
maxiter	Maximum number of iterations for the EM algorithm
norm.diff.conv	Convergence criterium for the parameters

norm.score.conv	Convergence criterium for the log-likelihood function
y.eps	Non-negative constant to be added to the original counts in y
mup	Weight for the constraints penalty part of the merit function
step	Interval length for the line search
printflag	If printflag=n the log-likelihood function is displayed any n iterations
old.tran.p	Starting values for the transition matrix
bb	Starting values for the observation probabilities
q.par	The percentage of parameters that must satisfy the convergence criterium, q.par has values in [0 1]

Details

Every column of y corresponds to an observed variable, every row in y reports the realization of the observed variable at each time occasion. The r realizations of each observed variable must be coded by the first r integers. The model.lat and model.obs are objects inheriting from class hmmmmod. So, the model for the transition matrix of the latent Markov chain and the model for the multinomial process can be marginal models specified by 'hmmm.model' or 'hmmm.model.X'. Consider a hidden Markov model for p observed variables and q latent variables. In defining the hmm models for observed and latent components using 'hmmm.model' bear in mind that: for the observations, the first q variables are the latent variables, followed by the p observed variables; in the latent model, the first q variables refer to the latent at time t, while the remaining q indicate the latent variables at time (t-1). Note that in both cases the first marginal set must contain only the latent variables. On the other hand, in defining the hmm models for the observations and latent chain using 'hmmm.model.X' consider that: for the observed model, the responses are the observed variables while the latent variables have the role of covariates and the number of their categories is declared in strata; in the latent model, the responses are the latent variables at time t, while the lagged variables at time (t-1) are considered as covariates. Declare the argument nlat only if model.obs is specified by 'hmmm.model' and model.lat is not explicitly defined.

Value

vecpar	List of two vectors of parameters of the observed and latent processes
model.obs	Information about the observed process
model.lat	Information about the latent process
initial	Invariant distribution of the latent process
Ptr	Estimated transition probabilities
Ptobs	Estimated observation probabilities
Ptr.iniz	Starting values of Ptr
Ptobs.iniz	Starting values for Ptobs
filter	Filtered probabilities
smooth	Smoothed probabilities
conv	List of convergence criteria.

Use 'print' or 'summary' to display the output.

References

Colombi R, Giordano S (2011) Lumpability for discrete hidden Markov models. Advances in Statistical Analysis, 95(3), 293-311.

See Also

print.hidden, summary.hidden, hmmm.model, hmmm.model.X

Examples

```
data(drinks)
y<-cbind(drinks$lemon.tea,drinks$orange.juice)
fm<-c("l-l-l")
fmargobs<-marg.list(fm)
#initial values of transition matrix and obs distribution given the two latent states
Ptr<-matrix(c(0.941, 0.199,0.059, 0.801),2,2,byrow=TRUE)
Ptobs<-matrix(c(0.053, 0.215, 0.206, 0.001, 0.039, 0.021, 0.020, 0.176, 0.270,
                0.000, 0.000, 0.000, 0.048, 0.263, 0.360, 0.065, 0.053, 0.211),
2,9,byrow=TRUE)
find<-~lat+lat*tea+lat*juice   # lat is the latent variable
model.obsf<-hmmm.model(marg=fmargobs,
lev=c(2,3,3),names=c("lat","tea","juice"),formula=find)

# model of independent observed variables given the latent states
modelind<-hidden.emfit(y,model.obsf,y.eps=0.01,maxit=10,maxiter=2500,
old.tran.p=Ptr,bb=Ptobs)
print(modelind,printflag=TRUE)

#alternative definition based on hmmm.model.X
f<-list(tea=~tea*lat,juice=~juice*lat,tea.juice="zero")
model.obsfX<-hmmm.model.X(marg=marg.list(c("l-l")),names=c("tea","juice"),
fnames=c("lat"),lev=c(3,3),strata=c(2))
modelindX<-hidden.emfit(y,model.obsfX,y.eps=0.01,maxit=10,maxiter=2500,
old.tran.p=Ptr,bb=Ptobs)
modelindX
summary(modelindX)
```

---

hmmm.chibar *chi-bar statistic test for hmm models*

Description

Function to calculate weights and pvalues of a chi-bar-square distributed statistic for testing hypotheses of inequality constraints on parameters of hmm models. The models in input are objects inheriting from class hmmmfit or mphfit.

Usage

```
hmmm.chibar(nullfit, disfit, satfit, repli = 6000,
kudo = FALSE, TESTAB = FALSE,
alpha = c(0.02,0.03,0), pesi = NULL)
```

Arguments

nullfit	The estimated model with inequalities turned into equalities
disfit	The estimated model with inequalities
satfit	The estimated model without inequalities
repli	Number of simulations
kudo	If TRUE, the chi-bar weights are not simulated but computed by the Kudo's method
TESTAB	If TRUE, the LR tuned testing procedure is performed (see Details)
alpha	Three significance levels c(alpha1, alpha2, alpha12) of the LR tuned testing procedure
pesi	The chi-bar weights if they are known

Details

All the 3 argument models must be obtained by 'hmmm.mlfit' or by 'mphineq.fit'. The method "Simulation 2" described in Silvapulle and Sen, 2005, pg. 79 is used if kudo = FALSE, otherwise the Kudo's exact method is used as described by El Barmi and Dykstra (1999). The Kudo's method can be reasonably used with less than 10-15 inequalities. If TESTA is the LR statistics for nullfit against the disfit model while TESTB is the LR statistics for disfit against the satfit model then the LR tuned testing procedure (Colombi and Forcina, 2013) runs as follows: accept nullfit if TESTB < y2 and TESTA < y1, where Pr(TESTB > y2) = alpha2-alpha12 and Pr(TESTA < y1, TESTB < y2) = 1-alpha1-alpha2, reject nullfit in favour of disfit if TESTA > y1 and TESTB < y12, where Pr(TESTA > y1, TESTB < y12) = alpha1, otherwise reject nullfit for satfit.

Value

A list with the statistics test of type A and B (Silvapulle and Sen, 2005, pg. 61) and their pvalues. If TESTAB = TRUE details on the LR tuned testing procedure (Colombi and Forcina, 2013) are reported.

References

Colombi R. Forcina A. (2013) Testing order restrictions in contingency tables. Submitted.

El Barmi H, Dykstra R (1999) Likelihood ratio test against a set of inequality constraints. Journal of Nonparametric Statistics, 11, 233-261.

Silvapulle MJ, Sen PK (2005) Constrained statistical inference, Wiley, New Jersey.

See Also

summary.hmmmchibar, print.hmmmchibar

Examples

```
data(polbirth)
# 1 = Politics; 2 = Birthcontrol
y<-getnames(polbirth,st=12,sep=";")
names<-c("Pol","Birth")
marglist<-c("l-m","m-l","l-l")
marginals<-marg.list(marglist,mflag="m")
ineq<-list(marg=c(1,2),int=list(c(1,2)),types=c("l","l"))

# definition of the model with inequalities on interactions in ineq
model<-hmmm.model(marg=marginals,dismarg=list(ineq),lev=c(7,4),names=names)

# saturated model
msat<-hmmm.mlfit(y,model)

# model with non-negative local log-odds ratios: "Likelihood ratio monotone dependence model"
mlr<-hmmm.mlfit(y,model,noineq=FALSE)

# model with null local log-odds ratios: "Stochastic independence model"
model0<-hmmm.model(marg=marginals,lev=c(7,4),sel=c(10:27),names=names)
mnull<-hmmm.mlfit(y,model0)

# HYPOTHESES TESTED:
#     testA --> H0=(mnull model) vs H1=(mlr model)
#     testB --> H0=(mlr model) vs H1=(msat model)

P<-hmmm.chibar(nullfit=mnull,disfit=mlr,satfit=msat)
summary(P)
```

`hmmm.mlfit` *fit a hmm model*

Description

Function to estimate a hierarchical multinomial marginal model.

Usage

```
hmmm.mlfit(y, model, noineq = TRUE, maxit = 1000,
norm.diff.conv = 1e-05, norm.score.conv = 1e-05,
y.eps = 0, chscore.criterion = 2,
m.initial = y, mup = 1, step = 1)
```

Arguments

`y`	A vector of frequencies of the contingency table
`model`	An object created by 'hmmm.model'
`noineq`	If TRUE inequality constraints specified in the model are ignored

maxit	Maximum number of iterations
norm.diff.conv	Convergence criterium value on the parameters
norm.score.conv	Convergence criterium value on the constraints
y.eps	Non-negative constant to be added to the original counts in y
chscore.criterion	If equal to zero, convergence information are printed at every iteration
m.initial	Initial estimate of m (expected frequencies)
mup	Weight for the constraints penalty part of the merit function
step	Interval length for the line search

Details

A sequential quadratic procedure is used to maximize the log-likelihood function under inequality and equality constraints. This function calls the procedure 'mphineq.fit' which is a generalization of the procedure 'mph.fit' by Lang (2004).

Value

An object of the class hmmmfit; an estimate of a marginal model defined by 'hmmm.model'. The output can be displayed using 'summary' or 'print'.

References

Bartolucci F, Colombi R, Forcina A (2007) An extended class of marginal link functions for modelling contingency tables by equality and inequality constraints. Statistica Sinica, 17, 691-711.

Bergsma WP, Rudas T (2002) Marginal models for categorical data. The Annals of Statistics, 30, 140-159.

Colombi R, Giordano S, Cazzaro M (2014) hmmm: An R Package for hierarchical multinomial marginal models. Journal of Statistical Software, 59(11), 1-25, URL http://www.jstatsoft.org/v59/i11/.

Lang JB (2004) Multinomial Poisson homogeneous models for contingency tables. The Annals of Statistics, 32, 340-383.

See Also

hmmm.model, hmmm.model.X, summary.hmmmfit, print.hmmmfit

Examples

```
data(relpol)
y<-getnames(relpol,st=12)
# 1 = Religion, 2 = Politics
names<-c("Rel","Pol")
marglist<-c("l-m","m-g","l-g")
marginals<-marg.list(marglist,mflag="m")

# Hypothesis of stochastic independence: all log odds ratios are null
model<-hmmm.model(marg=marginals,lev=c(3,7),sel=c(9:20),names=names)
```

```
fitmodel<-hmmm.mlfit(y,model)
print(fitmodel, aname="Independence model",printflag=TRUE)
summary(fitmodel)
```

---

hmmm.model *define a hmm model*

Description

Function to define a hierarchical multinomial marginal model.

Usage

```
hmmm.model(marg = NULL, dismarg = 0, lev, cocacontr = NULL, strata = 1,
Z = NULL, ZF = Z, X = NULL, D = NULL, E = NULL,
names = NULL, formula = NULL, sel = NULL)
```

Arguments

marg	A list of the marginal sets and their marginal interactions as described in Bartolucci et al. (2007). See below
dismarg	Similar to marg but used to define inequalities Kln(Am)>0. Default 0 if there are no inequalities
lev	Number of categories of the variables
cocacontr	A list of zero-one matrices to build "r" logits created by the function 'recursive'
strata	Number of strata defined by the combination of the categories of the covariates
Z	Zero-one matrix describing the strata
ZF	Zero-one matrix for strata with fixed number of observations
X	Design matrix for Cln(Mm)=Xbeta. Identity matrix if not declared. It can be defined later or changed only by using the function 'create.XMAT'
D	If the matrix D is declared, the inequalities are expressed as DKln(Am)>0. Useful for changing the sign of inequalities or for selecting a subset of inequalities
E	If E is a matrix, then E defines the equality contrasts as ECln(Mm)=0
names	A character vector whose elements are the names of the variables
formula	Formula of the reference log-linear model
sel	Vector reporting the positions of the interactions constrained to be zero

Details

Variables are denoted by integers, the lower the number identifying the variable the faster its category subscript changes in the vectorized contingency table. Suppose that the variables are 1 and 2 with categories k_1, k_2, the joint frequencies y = y_ij, where i=1,...,k_1, j=1,...,k_2, are arranged in a vector so that the subscript i changes faster than j. If `strata` is greater than one, the vectorized contingency tables must be entered strata by strata. So that, for example, if the variables are distinguished in responses and covariates, the categories of the covariates determine the strata and the data are arranged in such a way that the categories of the response variable changes faster than the categories of covariate. The names of the variables in `names` must be declared according to the order of the variables.

The list `marg` of the marginal sets of a complete hierarchical marginal parameterization, together with the types of logits for the variables, must be created by the function 'marg.list'. See the help of this function for more details. If `marg` is not specified the multivariate logit model by Glonek and McCullagh (1995) with interactions of type local is used. The list `marg` is used to create the link function Cln(Mm) and its derivative (m is the vector of expected frequencies).

If the model is defined in the form Cln(Mm) = Xbeta, the matrix X has to be declared (see the function 'create.XMAT'). If there are only nullity constraints on parameters, the model is in the form ECln(Mm)=0 and X is ignored. In such a case, E can be declared as matrix or it is automatically constructed if `sel` is declared. If `sel` is not NULL, then the model is defined under equality constraints, i.e. ECln(Mm)=0. When `X`, `E` and `sel` are left at default level, a saturated model is defined.

For models with inequality constraints on marginal parameters, the input argument `dismarg` is declared as a list whose components are of type: `list(marg=c(1,2),int=list(c(1),c(1,2)), types=c("g","l"))`, with elements `marg`: the marginal set, `int`: the list of the interaction subject to inequality constraint, and `types`: the logit used for every variable ("g"=global, "l"=local, "c"=continuation, "rc"=reverse continuation, "r"=recursive, "b"=baseline, "marg" is assigned to each variable not belonging to the marginal set). This list is used to create the link function Cln(Mm) and its derivative for the inequality constraints.

The matrix Z is of dimension c x s, where c is the number of counts and s is the number of strata or populations. Thus, the rows correspond to the number of observations and the columns correspond to the strata. A 1 in row i and column j means that the ith count comes from the jth stratum. Note that Z has exactly one 1 in each row, and at least one 1 in each column. When the population matrix Z is a column vector of 1 indicates that all the counts come from the same and only stratum. For hmm models, it is assumed that all the strata have the same number of response levels. If Z is not given, a population Z matrix corresponding to data entered by `strata` is defined and ZF=Z. For non-zero ZF, the columns are a subset of the columns in Z. If the jth column of Z is included in ZF, then the sample size of the jth stratum is considered fixed, otherwise if the jth column of Z is NOT included in ZF, the jth stratum sample size is taken to be a realization of a Poisson random variable. As ZF=Z the sample size in every stratum is fixed; this is the (product-)multinomial setting.

The formula of the reference log-linear model must be defined using the names of the variables declared in `names`, for example `names<-c("A","B","C","D")`, `formula=~A*C*D+B*C*D+A:B`. The interactions not involved in `formula` cannot be further constrained in the marginal model. The default `formula = NULL` indicates the saturated log-linear model as reference model. The likelihood function of the reference model is maximized by 'hmmm.mlfit' under the constraints ECln(Mm)=0 on the marginal parameters.

The arguments `dismarg` and `formula` can be used only if `strata=1`.

Value

An object of the class hmmmmod; it describes a marginal model that can be estimated by 'hmmm.mlfit'.

References

Bartolucci F, Colombi R, Forcina A (2007) An extended class of marginal link functions for modelling contingency tables by equality and inequality constraints. Statistica Sinica, 17, 691-711.

Bergsma WP, Rudas T (2002) Marginal models for categorical data. The Annals of Statistics, 30, 140-159.

Cazzaro M, Colombi R (2009) Multinomial-Poisson models subject to inequality constraints. Statistical Modelling, 9(3), 215-233.

Colombi R, Giordano S, Cazzaro M (2014) hmmm: An R Package for hierarchical multinomial marginal models. Journal of Statistical Software, 59(11), 1-25, URL http://www.jstatsoft.org/v59/i11/.

Glonek GFV, McCullagh P (1995) Multivariate logistic models for contingency tables. Journal of the Royal Statistical Society, B, 57, 533-546.

See Also

hmmm.model.X, create.XMAT, summary.hmmmmod, print.hmmmmod, marg.list, recursive, hmmm.mlfit

Examples

```
data(madsen)
# 1 = Influence; 2 = Satisfaction;  3 = Contact; 4 = Housing
names<-c("Inf","Sat","Co","Ho")
y<-getnames(madsen,st=6)

# hmm model -- marginal sets: {3,4} {1,3,4} {2,3,4} {1,2,3,4}
margi<-c("m-m-l-l","l-m-l-l","m-l-l-l","l-l-l-l")
marginals<-marg.list(margi,mflag="m")
model<-hmmm.model(marg=marginals,lev=c(3,3,2,4),names=names)
summary(model)

# hmm model with equality constraints
# independencies 1_||_4|3 and 2_||_3|4 impose equality constraints
sel<-c(12:23,26:27,34:39) # positions of the zero-constrained interactions
model_eq<-hmmm.model(marg=marginals,lev=c(3,3,2,4),sel=sel,names=names)
summary(model_eq)

# hmm model with inequality constraints
# the distribution of 1 given 4 is stochastically decreasing wrt the categories of 3;
# the distribution of 2 given 3 is stochastically decreasing wrt the categories of 4:
marg134ineq<-list(marg=c(1,3,4),int=list(c(1,3)),types=c("l","marg","l","l"))
marg234ineq<-list(marg=c(2,3,4),int=list(c(2,4)),types=c("marg","l","l","l"))
ineq<-list(marg134ineq,marg234ineq)
model_ineq<-hmmm.model(marg=marginals,lev=c(3,3,2,4),dismarg=ineq,D=diag(-1,8),names=names)
summary(model_ineq)
# The argument D is used to turn the 8 inequalities from
# non-negative (default) into non-positive constraints
```

`hmmm.model.X` *hmm model with covariates effect on parameters*

Description

Function to define a hmm model whose parameters depend on covariates.

Usage

```
hmmm.model.X(marg, lev, names, Formula = NULL, strata = 1,
fnames = NULL, cocacontr = NULL, ncocacontr = NULL, replace=TRUE)
```

Arguments

`marg`	A list of the marginal sets and their marginal interactions as described in Bartolucci et al. (2007). See details of `hmmm.model`
`lev`	Number of categories of the response variables
`names`	A character vector whose elements are the names of the response variables
`Formula`	List of model-formulas; one formula for every marginal interaction
`strata`	Number of categories of the covariates that describe the strata
`fnames`	Names of the covariates that describe the strata
`cocacontr`	A list of zero-one matrices to build "r" logits created by the function 'recursive'
`ncocacontr`	Number of contrasts for every covariate, if NULL the maximum number is used
`replace`	If TRUE a new model object with design matrix X is produced, if FALSE the list of design matrices associated to each element specified in Formula is returned

Details

The arguments `names` and `fnames` report the names of responses and covariates according to the order in which the variables are declared, see details of function 'hmmm.model'.

When the marginal interactions of a hmm model are defined in terms of a linear predictor of covariates Cln(Mm)=Xbeta, the list of model formulas defines additive effects of covariates on the interactions. In a case with two response variables declared by `names<-c("A","B")` and two covariates, named C and D by `fnames=c("C","D")`, the additive effect of the covariates on marginal logits of A and B and log odds ratios (A.B) of the two responses is specified by the following `Formula`: `Formula<-list(A=~A*(C+D), B=~B*(C+D), A.B=~A.B*(C+D))`. Use "zero" to constrain to zero all the interactions of a given type. The saturated model is the default if `Formula` is not specified.

Value

An object of the class `hmmmmod`; it describes a marginal model with effects of covariates on the interactions. This model can be estimated by 'hmmm.mlfit'.

References

Colombi R, Giordano S, Cazzaro M (2014) hmmm: An R Package for hierarchical multinomial marginal models. Journal of Statistical Software, 59(11), 1-25, URL http://www.jstatsoft.org/v59/i11/.

Glonek GFV, McCullagh P (1995) Multivariate logistic models for contingency tables. Journal of the Royal Statistical Society, B, 57, 533-546.

Marchetti GM, Lupparelli M (2011) Chain graph models of multivariate regression type for categorical data. Bernoulli, 17, 827-844.

See Also

hmmm.model, create.XMAT, summary.hmmmmod, print.hmmmmod, marg.list, recursive, hmmm.mlfit

Examples

```
data(accident)
y<-getnames(accident,st=9,sep=";")
# responses: 1 = Type, 2 = Time; covariates: 3 = Age, 4 = Hour

marginals<-marg.list(c("b-marg","marg-g","b-g"))
al<-list(
Type=~Type*(Age+Hour),
Time=~Time*(Age+Hour),
Type.Time=~Type.Time*(Age+Hour)
)
# model with additive effect of the covariates on logits and log-o.r. of the responses
model<-hmmm.model.X(marg=marginals,lev=c(3,4),names=c("Type","Time"),
Formula=al,strata=c(3,2),fnames=c("Age","Hour"))
mod<-hmmm.mlfit(y,model,y.eps=0.1)
```

---

inv_GMI *function to compute a vector of joint probabilities from a vector of generalized marginal interactions (gmi)*

Description

Given an hmmm model and the vector of its generalized interactions eta, the vector of joint probabilities p is computed by inverting eta=C*ln(M*p)

Usage

```
inv_GMI(etpar, mod, start = rep(0, prod(mod$modello$livelli)))
```

Arguments

etpar	Vector of gmi
mod	hmmm model corresponding to etapar; an object of class hmmmod created by hmmm.model
start	Starting values for log-linear parameters in the non linear equations problem

Value

Vector of joint probabilities

See Also

GMI, hmmm.model

Examples

```
# a joint distribution of 2 variables with 4 categories each

p4<-c(
0.0895, 0.0351 ,0.0004, 0.0003,
0.0352, 0.2775, 0.0619, 0.0004,
0.0004, 0.0620, 0.2775, 0.0351,
0.0001, 0.0004, 0.0352, 0.089)

marg<-marg.list(c("l-m","m-l","l-l"), mflag="m")
labelrisp<-c("R1","R2")
modello<-hmmm.model(marg=marg,lev=c(4,4),names=labelrisp)

etpar<-GMI(c(p4),c("l-m","m-l","l-l"),c(4,4),labelrisp,mflag="m")
etpar$gmi
p4rec<-inv_GMI(etpar$gmi,modello)
P<-cbind(p4rec,c(p4),c(p4)-p4rec)
colnames(P)<-c("prob","prob from eta","check")
P
```

---

kentucky *Kentucky traffic accident data*

Description

The traffic accident data collected by the Kentucky State Police from 1995 to 1999. The annual numbers of vehicle occupants involved in Kentucky accidents are classified according to 3 variables: injury, restraint usage and year.

Usage

```
data(kentucky)
```

Format

A data frame whose columns contain:

`Injury` A factor with levels: 1 = not injured; 2 = possible injury; 3 = nonincapacitating injury; 4 = incapacitating injury; 5 = killed

`Restraint.usage` A factor with levels: `yes` = restraint used, `no` = restraint not used

`Year` 1995, 1996, 1997, 1998, 1999

`Freq` A numeric vector of frequencies

Source

www.kentuckystatepolice.org/text/data.htm

References

Lang JB (2005) Homogeneous linear predictor models for contingency tables. Journal of the American Statistical Association, 100, 121-134.

Examples

```
data(kentucky)
```

`loglin.model` *define a log-linear model*

Description

Function to specify a hierarchical log-linear model. This is a particular case of a hmm model.

Usage

```
loglin.model(lev, int = NULL, strata = 1, dismarg = 0, type = "b",
D = TRUE, c.gen = TRUE, printflag = FALSE, names = NULL, formula = NULL)
```

Arguments

`lev`	Vector of number of categories of variables
`int`	Generating class of the log-linear model (must be a list) or list of all the interactions included
`strata`	Number of strata
`dismarg`	List of interactions constrained by inequalities - see 'hmmm.model'
`type`	"b" for baseline logits, "l" for local logits
`D`	Input argument for inequalities - see 'hmmm.model'
`c.gen`	If FALSE the input int must be the list of the minimal interaction sets to be excluded
`printflag`	If TRUE information on the included and excluded interactions are given
`names`	A character vector whose elements are the names of the variables
`formula`	A formula describing a log-linear model

Details

This function simplifies 'hmmm.model' in the case of log-linear models. If `formula` is employed, `c.gen` and `int` must not be declared while `names` must be specified.

Value

An object of the class hmmmmod defining a log-linear model that can be estimated by 'hmmm.mlfit'.

Note

If int and formula are not supplied a saturated log-linear model is defined. For log-linear models where the parameters depend on covariates first define a saturated log-linear model and then use the function 'create.XMAT'.

References

Agresti A (2012) Categorical data Analysis, (3ed), Wiley, New York.

Bergsma W, Croon M, Hagenaars JA (2009) Marginal Models for Dependent, Clustered, and Longitudinal Categorical Data. Springer.

See Also

hmmm.model, hmmm.mlfit, create.XMAT

Examples

```
data(madsen)
y<-getnames(madsen)
names<-c("Infl","Sat","Co","Ho")

f<-~Co*Ho+Sat*Co+Infl*Co+Sat*Ho+Infl*Sat
model<-loglin.model(lev=c(3,3,2,4),formula=f,names=names)

# alternatively
# model<-loglin.model(lev=c(3,3,2,4),
# int=list(c(3,4),c(2,3),c(1,3),c(2,4),c(1,2)),names=names)

mod<-hmmm.mlfit(y,model,maxit=3000)
print(mod,printflag=TRUE)
```

---

madsen *Madsen data*

Description

The dataset concerns 1681 rental property residents classified according to their satisfaction from the house, perceived influence on the management of the property, type of rental accommodation, and contact with other residents.

Usage

```
data(madsen)
```

Format

A data frame whose columns contain:

`Influence` A factor with levels: `low`, `medium`, `high`

`Satisfaction` A factor with levels: `low`, `medium`, `high`

`Contact` A factor with levels: `low`, `high`

`Housing` A factor with levels: `tower block`, `apartment`, `atrium house`, `terraced house`

`Freq` A numeric vector of frequencies

References

Madsen M (1976) Statistical analysis of multiple contingency tables. Two examples. Scandinavian Journal of Statistics, 3, 97-106.

Examples

```
data(madsen)
```

`marg.list` *lists of marginal sets*

Description

An easy option to define the first input argument `marg` of the function 'hmmm.model' which specifies the list of marginal sets of a hmm model.

Usage

```
marg.list(all.m, sep = "-", mflag = "marg")
```

Arguments

`all.m`	A character vector with one element for every marginal set, see below
`sep`	The separator used between logits type, default "-"
`mflag`	The symbol used to denote variables that are marginalized, default "marg"

Details

`all.m` is a string indicating the logit types used to build the interactions in each marginal set. For each variable in the marginal set the corresponding logit symbol is inserted ("b" baseline, "g" global, "c" continuation, "rc" reverse continuation, "r" recursive, "l" local). Symbols are separated by `sep` and the variables not included in the marginal set are denoted by `mflag`. So, for example, "marg-g-c" indicates a marginal set involving variables 2, 3 with global and continuation logits respectively.

Value

The list `marg` used as first input argument in 'hmmm.model' – see the function 'hmmm.model'.

Note

This function creates the complete list of the interactions that can be defined in a marginal set. Therefore, it cannot be used to specify only the interactions subject to inequality constraints. When inequalities are involved in the model, marginal sets and types of logits are declared as illustrated in the details of function 'hmmm.model'.

References

Colombi R, Giordano S, Cazzaro M (2014) hmmm: An R Package for hierarchical multinomial marginal models. Journal of Statistical Software, 59(11), 1-25, URL http://www.jstatsoft.org/v59/i11/.

See Also

hmmm.model

Examples

```
data(madsen)
marginals<-c("m-m-b-b","g-m-b-b","m-g-b-b","g-g-b-b")
margi<-marg.list(marginals,mflag="m")
names<-c("Inf","Sat","Co","Ho")
model<-hmmm.model(marg=margi,lev=c(3,3,2,4),names=names)
print(model)
```

---

mphineq.fit *fit mph models under inequality constraints*

Description

Function to maximize the log-likelihood function of multinomial Poisson homogeneous (mph) models under nonlinear equality and inequality constraints.

Usage

```
mphineq.fit(y, Z, ZF = Z, h.fct = 0, derht.fct = 0, d.fct = 0,
derdt.fct = 0, L.fct = 0, derLt.fct = 0, X = NULL, formula = NULL,
names = NULL, lev = NULL, E = NULL, maxiter = 100,
step = 1, norm.diff.conv = 1e-05, norm.score.conv = 1e-05,
y.eps = 0, chscore.criterion = 2, m.initial = y, mup = 1)
```

Arguments

y	Vector of frequencies of the multi-way table
Z	Population matrix. The population matrix Z is a c x s zero-one matrix, where c is the number of counts and s is the number of strata or populations. Thus, the rows correspond to the number of observations and the columns correspond to the strata. A 1 in row i and column j means that the ith count comes from the

	jth stratum. Note that Z has exactly one 1 in each row, and at least one 1 in each column. When Z = matrix(1,length(y),1) is a column vector of 1, all the counts come from the same and only stratum.
ZF	Sample constraints matrix. For non-zero ZF, if the columns are a subset of the columns in the population matrix Z, then the sample size of the jth stratum is considered fixed, otherwise if the jth column of Z is NOT included in ZF, the jth stratum sample size is taken to be a realization of a Poisson random variable. When ZF=0, all of the stratum sample sizes are taken to be realizations of Poisson random variables. The default, ZF=Z, means that all the stratum sample sizes are fixed; this is the (product-)multinomial setting. Note that ZF'y = n is the vector of fixed sample sizes
h.fct	Function h(m) of equality constraints, m is the vector of expected frequencies. This function of m must return a vector
derht.fct	Derivative of h(m), if not supplied numerical derivative are used
d.fct	Function for inequality constraints d(m)>0. This function of m must return a vector
derdt.fct	Derivative of d(m), if not supplied numerical derivative are used
L.fct	Link function for the linear model L(m)=Xbeta
derLt.fct	Derivative of L(m), if not supplied numerical derivative are used
X	Model matrix for L(m)=Xbeta
formula	Formula of the reference log-linear model
names	A character vector whose elements are the names of the variables
lev	Number of categories of the variables
E	If E is a matrix, then X is ignored and E defines the equality contrasts as EL(m)=0
maxiter	Maximum number of iterations
step	Interval length for the linear search
norm.diff.conv	Convergence criterium for parameters
norm.score.conv	Convergence criterium for constraints
y.eps	Non-negative constant to be temporarily added to the original frequencies in y
chscore.criterion	If zero, convergence information are printed at every iteration
m.initial	Initial estimate of m
mup	Weight for the constraint part of the merit function

Details

This function extends 'mph.fit' written by JB Lang, Dept of Statistics and Actuarial Science University of Iowa, in order to include inequality constraints. In particular, the Aitchison Silvey (AS) algorithm has been replaced by a sequential quadratic algorithm which is equivalent to AS when inequalities are not present. The R functions 'quadprog' and 'optimize' have been used to implement the sequential quadratic algorithm. More precisely, the AS updating formulas are replaced by an equality-inequality constrained quadratic programming problem. The 'mph.fit' step halving linear search is replaced by an optimal step length search performed by 'optimize'.

References

Colombi R, Giordano S, Cazzaro M (2014) hmmm: An R Package for hierarchical multinomial marginal models. Journal of Statistical Software, 59(11), 1-25, URL http://www.jstatsoft.org/v59/i11/.

Lang JB (2004) Multinomial Poisson homogeneous models for contingency tables. The Annals of Statistics, 32, 340-383.

Lang JB (2005) Homogeneous linear predictor models for contingency tables. Journal of the American Statistical Association, 100, 121-134.

See Also

print.mphfit, summary.mphfit, hmmm.model, hmmm.mlfit

Examples

```
y <- c(104,24,65,76,146,30,50,9,166) # Table 2 (Lang, 2004)
y <- matrix(y,9,1)

# population matrix: 3 strata with 3 observations each
Z <- kronecker(diag(3),matrix(1,3,1))
# the 3rd stratum sample size is fixed
ZF <- kronecker(diag(3),matrix(1,3,1))[,3]

#########################################################################
# Let (i,j) be a cross-citation, where i is the citing journal and j is
# the cited journal. Let m_ij be the expected counts of cross-citations.
# The Gini concentrations of citations for each of the journals are:
# G_i = sum_j=1_3 (m_ij/m_i+)^2  for i=1,2,3.


Gini<-function(m) {
 A<-matrix(m,3,3,byrow=TRUE)
 GNum<-rowSums(A^2)
 GDen<-rowSums(A)^2
 G<-GNum/GDen
 c(G[1],G[2],G[3])-c(0.410,0.455,0.684)
 }


# h_1 = c(G1,G2,G3)-c(0.410,0.455,0.684) = 0
# HYPOTHESIS: no change in Gini concentrations
# from the 1987-1989 observed values

mod_eq <- mphineq.fit(y,Z,ZF,h.fct=Gini)

print(mod_eq)

# Example of MPH model subject to inequality constraints

# d_1 = c(G1,G2,G3)-c(0.410,0.455,0.684) >= 0
# HYPOTHESIS: increase in Gini concentrations
# from the 1987-1989 observed values
```

```
mod_ineq <- mphineq.fit(y,Z,ZF,d.fct=Gini)


# Reference model: model without inequalities --> saturated model
mod_sat <-mphineq.fit(y,Z,ZF)


# HYPOTHESES TESTED:
# NB: testA --> H0=(mod_eq) vs H1=(mod_ineq model)
# testB --> H0=(mod_ineq model) vs H1=(sat_mod model)

hmmm.chibar(nullfit=mod_eq,disfit=mod_ineq,satfit=mod_sat)
```

`polbirth` *political orientation and teenage birth control data*

Description

Data on political orientation and opinion on teenage birth control of a sample of 911 U.S. citizens.

Usage

```
data(polbirth)
```

Format

A data frame whose columns contain:

`Politics` A factor with levels: `Extremely liberal`, `Liberal`, `Slightly liberal`, `Moderate`, `Slightly conservative`, `Conservative`, `Extremely conservative`

`Birthcontrol` A factor with levels: `Strongly agree`, `Agree`, `Disagree`, `Strongly disagree`

`Freq` A numeric vector of frequencies

Details

This is a sub-data frame obtained by marginalizing the data frame 'relpolbirth' with respect to the variable religion.

Source

General Social Survey, 1993.

References

Bergsma W, Croon M, Hagenaars JA (2009) Marginal Models for Dependent, Clustered, and Longitudinal Categorical Data. Springer.

Examples

```
data(polbirth)
```

`print.hidden` *print for the class hidden*

Description

The generic function 'print' is adapted to the objects inheriting from class `hidden` (print.hidden) to display the results of the estimation of a hidden model by 'hidden.emfit'.

Usage

```
## S3 method for class 'hidden'
print(x,printflag = FALSE,...)
```

Arguments

`x`	An object of the class `hidden`, i.e. a result of 'hidden.emfit'
`printflag`	If TRUE the estimated parameters are displayed
`...`	Further arguments passed to or from other methods

Value

The value of the log-likelihood functions and the estimated parameters of transition and observation models are printed.

See Also

`hidden.emfit`, `summary.hidden`

`print.hmmmchibar` *print for hmmm.chibar*

Description

Function to print the results for tests of type A and B (Silvapulle and Sen, 2005) on inequality constraints. The generic function 'print' is adapted to the objects inheriting from class `hmmmchibar` (print.hmmmchibar).

Usage

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

Arguments

x	An object of the class hmmmchibar
...	Further arguments passed to or from other methods

Value

It provides the output of the function 'hmmm.chibar', that is the results of testing models defined by 'hmmm.model' and estimated by 'hmmm.mlfit', with equality and inequality constraints on marginal interactions. The statistics test of type A and B and their pvalues are tabulated. If TESTAB = TRUE when the function 'hmmm.chibar' is called, the output of the LR tuned testing procedure (Colombi and Forcina, 2013) is displayed.

Note

Use 'summary' to display a much detailed output.

References

Colombi R. Forcina A. (2013) Testing order restrictions in contingency tables. Submitted

Silvapulle MJ, Sen PK (2005) Constrained statistical inference, Wiley, New Jersey.

See Also

hmmm.chibar, summary.hmmmchibar

---

print.hmmmfit *print for the class hmmmfit*

Description

The generic function 'print' is adapted to the objects inheriting from class hmmmfit (print.hmmmfit) to display the results of the estimation of a hmm model by 'hmmm.mlfit'.

Usage

```
## S3 method for class 'hmmmfit'
print(x, aname = " ", printflag = FALSE, ...)
```

Arguments

x	An object of the class hmmmfit, i.e. a result of 'hmmm.mlfit'
aname	The name of the fitted object model
printflag	If FALSE only the goodness-of-fit test is displayed, if TRUE the estimates of the interaction parameters are also returned
...	Further arguments passed to or from other methods

Details

The output provides the likelihood ratio statistic test to assess the fitting of the model estimated by 'hmmm.mlfit'. Degrees of freedom and pvalues are meaningful only for the hmm models without inequality constraints (see 'hmmm.chibar' to test hmm models defined under inequality constraints on interactions). Moreover, if `printflag` is TRUE, the estimated interactions are displayed for every stratum, together with the marginal sets where they are defined and the type of logits considered.

Note

Use 'summary' to display a much detailed output.

See Also

`hmmm.mlfit`, `summary.hmmmfit`, `anova.hmmmfit`

Examples

```
data(relpol)
y<-getnames(relpol,st=12)
# 1 = Religion, 2 = Politics
names<-c("Rel","Pol")
marglist<-c("1-m","m-g","1-g")
marginals<-marg.list(marglist,mflag="m")

# Hypothesis of stochastic independence: all log odds ratios are null
model<-hmmm.model(marg=marginals,lev=c(3,7),sel=c(9:20),names=names)
fitmodel<-hmmm.mlfit(y,model)
print(fitmodel,aname="independence model",printflag=TRUE)
# summary(fitmodel)
```

---

recursive *recursive marginal interactions*

Description

A function to define logits of recursive (or nested) type.

Usage

```
recursive(...)
```

Arguments

`...`	As many inputs as there are variables in the multi-way table. Each input is a matrix of values -1,0,1 to define recursive logits or 0 for logits of different type

Details

This function is used when logits of type "r" are used for at least one variable. An input argument for each categorical variable is necessary. Inputs are ordered according to the order of the variables.

For a categorical variable with k categories, k-1 recursive logits can be defined using a matrix with k-1 rows and k columns. The rows of this matrix specify the categories whose probabilities constitute numerator and denominator of every recursive logit. Specifically, in every row, a value among -1,0,1 is associated to every category: value 1 (-1) corresponds to the category whose probability is cumulated at the numerator (denominator), 0 if the category is not involved.

Value

A zero-one matrix to be assigned to the `cocacontr` input argument in defining a model by 'hmmm.model' when logits "r" are used for at least one variable in the multi-way table.

References

Cazzaro M, Colombi R (2008) Modelling two way contingency tables with recursive logits and odds ratios. Statistical Methods and Applications, 17, 435-453.

Cazzaro M, Colombi R (2013) Marginal nested interactions for contingency tables. Communications in Statistics - Theory and Methods, to appear.

Colombi R, Giordano S, Cazzaro M (2014) hmmm: An R Package for hierarchical multinomial marginal models. Journal of Statistical Software, 59(11), 1-25, URL http://www.jstatsoft.org/v59/i11/.

See Also

`hmmm.model`, `create.XMAT`, `hmmm.model.X`

Examples

```
data(kentucky)
# 1 = injury 2 = restraint 3 = year
y<-getnames(kentucky,st=4)

marglist<-marg.list(c("m-m-l","m-l-l","r-l-l"),mflag="m")
R1<-matrix(c(1,1,1,-1,-1,
             0,0,0,1,-1,
             1,1,-1,0,0,
             1,-1,0,0,0),4,5,byrow=TRUE)
# logits of recursive (or nested) type for variable 1:
# log p(injury<=3)/p(injury>3); log p(injury=4)/p(injury=5);
# log p(injury<=2)/p(injury=3); log p(injury=1)/p(injury=2);
rec<-recursive(R1,0,0) # only variable 1 has recursive logits

# additive effect of variables 2,3 on the recursive logits of variable 1
model<-hmmm.model(marg=marglist,lev=c(5,2,5),sel=c(34:49),cocacontr=rec)
mod<-hmmm.mlfit(y,model)
print(mod,printflag=TRUE)
```

relpol *religion and political orientation data*

Description

Data on religion and political orientation of a sample of 911 U.S. citizens.

Usage

```
data(relpol)
```

Format

A data frame whose columns contain:

`Religion` A factor with levels: `Protestant`, `Catholic`, `None`

`Politics` A factor with levels: `Extremely liberal`, `Liberal`, `Slightly liberal`, `Moderate`, `Slightly conservative`, `Conservative`, `Extremely conservative`

`Freq` A numeric vector of frequencies

Details

This is a sub-data frame obtained by marginalizing the data frame 'relpolbirth' with respect to the variable opinion on teenage birth control.

Source

General Social Survey, 1993

References

Bergsma W, Croon M, Hagenaars JA (2009) Marginal models for dependent, clustered, and longitudinal categorical data. Springer.

Examples

```
data(relpol)
```

relpolbirth *religion, political orientation and teenage birth control data*

Description

Data on religion, political orientation and opinion on teenage birth control of a sample of 911 U.S. citizens.

Usage

```
data(relpolbirth)
```

Format

A data frame whose columns contain:

`Religion` A factor with levels: `Protestant`, `Catholic`, `None`

`Politics` A factor with levels: `Extremely liberal`, `Liberal`, `Slightly liberal`, `Moderate`, `Slightly conservative`, `Conservative`, `Extremely conservative`

`Birthcontrol` A factor with levels: `Strongly agree`, `Agree`, `Disagree`, `Strongly disagree`

`Freq` A numeric vector of frequencies

Source

General Social Survey, 1993

References

Bergsma W, Croon M, Hagenaars JA (2009) Marginal models for dependent, clustered, and longitudinal categorical data. Springer.

Examples

```
data(relpolbirth)
```

summary.hidden *summary for the class hidden*

Description

The generic function 'summary' is adapted to the objects inheriting from class `hidden` (summary.hidden) to display the results of the estimation of a hidden model by 'hidden.emfit'.

Usage

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

Arguments

object	An object of the class hidden, i.e. a result of 'hidden.emfit'
...	Further arguments passed to or from other methods

Value

The transition probabilities and the probabilities of observations given the latent states are printed.

Note

Use 'print' to display the estimated parameters.

See Also

hidden.emfit, print.hidden

---

summary.hmmmchibar *summary for hmmm.chibar*

Description

Function to print the results for tests of type A and B (Silvapulle and Sen, 2005) on inequality constraints and to tabulate the chi-bar distribution functions of the statistics test. The generic function 'summary' is adapted to the objects inheriting from class hmmmchibar (summary.hmmmchibar).

Usage

```
## S3 method for class 'hmmmchibar'
summary(object, plotflag = 1, step = 0.01, lsup = 0,...)
```

Arguments

object	An object of the class hmmmchibar
plotflag	1 to print only pvalues and statistic values, 2 to display the survival functions for type A and type B statistics tests and 3 to provide a plot of the survival functions (red: type B, black: type A)
step	Distance between points at which the distribution functions are evaluated
lsup	Distribution functions are evaluated in the interval 0 - lsup
...	Further arguments passed to or from other methods

Value

It provides the output of the function 'hmmm.chibar', that is the results of testing models defined by 'hmmm.model' and estimated by 'hmmm.mlfit', with equality and inequality constraints on marginal interactions. The statistics test of type A and B, their pvalues and the chi-bar distribution functions are tabulated. If `TESTAB = TRUE` when the function 'hmmm.chibar' is called, the output of the LR tuned testing procedure (Colombi and Forcina, 2013) is displayed.

Note

Use 'print' for a short output.

References

Colombi R. Forcina A. (2013) Testing order restrictions in contingency tables. Submitted

Silvapulle MJ, Sen PK (2005) Constrained statistical inference, Wiley, New Jersey.

See Also

`hmmm.chibar`, `print.hmmmchibar`

`summary.hmmmfit` *summary for the class hmmmfit*

Description

The generic function 'summary' is adapted to the objects inheriting from class `hmmmfit` (summary.hmmmfit) to display the results of the estimation of a hmm model by 'hmmm.mlfit'.

Usage

```
## S3 method for class 'hmmmfit'
summary(object, cell.stats = TRUE, ...)
```

Arguments

`object`	An object of the class `hmmmfit`, i.e. a result of 'hmmm.mlfit'
`cell.stats`	If TRUE cell-specific statistics are returned
`...`	Further arguments passed to or from other methods

Details

The marginal interactions of a hmm model can be defined in terms of linear predictor of covariates Cln(Mm)=Xbeta, where the X matrix is specified by 'create.XMAT' and the parameters beta indicate the additive effects of covariate on the marginal interactions. The function 'hmmm.mlfit' estimates either the parameters beta and the interactions; the function 'summary' of a fitted model (by 'hmmm.mlfit') returns the estimated betas and the estimated interactions, while the function 'print' provides the estimated interactions only. If the model is defined under equality constraints ECln(Mm)=0, parameters betas are meaningless so they are not printed.

The output of 'summary' provides: 1. values of the likelihood ratio and Pearson's score statistics, degrees of freedom and pvalues. Note that degrees of freedom and pvalues are meaningful only for the hmm models without inequality constraints (see 'hmmm.chibar' to test hmm models defined under inequality constraints on interactions); 2. the linear predictor model results: estimated betas, standard errors, z-ratios, pvalues; estimated interactions, standard errors, residuals; 3. cell-specific statistics: observed and predicted frequencies of the multi-way table, estimated joint probabilities with standard errors, adjusted residuals; 4. convergence statistics.

Note

Use 'print' to display only the goodness-of-fit test and the estimated interactions.

See Also

hmmm.mlfit, print.hmmmfit, anova.hmmmfit, create.XMAT

Examples

```
data(relpol)
y<-getnames(relpol,st=12,sep=";")
# 1 = Religion, 2 = Politics
names<-c("Rel","Pol")
marglist<-c("l-m","m-g","l-g")
marginals<-marg.list(marglist,mflag="m")

# Hypothesis of stochastic independence: all log odds ratios are null
model<-hmmm.model(marg=marginals,lev=c(3,7),sel=c(9:20),names=names)
fitmodel<-hmmm.mlfit(y,model)
# print(fitmodel,aname="Independence model",printflag=TRUE)
summary(fitmodel)
```

---

summary.hmmmmod *summary and print for the class hmmmmod*

Description

The generic functions 'summary' and 'print' are adapted to the objects inheriting from class hmmmod (summary.hmmmmod, print.hmmmmod) to display the summary of a model defined by 'hmmm.model'.

Usage

```
## S3 method for class 'hmmmmod'
summary(object,...)
## S3 method for class 'hmmmmod'
print(x,...)
```

Arguments

object,x	An object of the class hmmmod, i.e. a result of 'hmmm.model'
...	Further arguments passed to or from other methods

Details

The output provides the list of interactions and the marginal distributions where those interactions are defined. The names of the involved variables are displayed if names is not NULL. For every interaction, the logit type used for each variable in the interaction set and the number of parameters are indicated. The last two columns give the position of the parameters in the vector where all the interactions are arranged.

Note

Functions 'summary' and 'print' display the same output.

See Also

hmmm.model

Examples

```
marginals<-marg.list(c("g-m","m-l","g-l"),mflag="m")
model<-hmmm.model(marg=marginals,lev=c(3,7),names=c("A","B"))
summary(model) # or print(model)
```

---

summary.mphfit *summary and print for the class* mphfit

Description

The generic functions 'summary' and 'print' are adapted to the objects inheriting from class mphfit (summary.mphfit, print.mphfit) to display the results of the estimation of a mph model by 'mphineq.fit'.

Usage

```
## S3 method for class 'mphfit'
summary(object, ...)
## S3 method for class 'mphfit'
print(x,...)
```

Arguments

`object,x`	An object inheriting from class `mphfit`, i.e. a result of 'mphineq.fit'
`...`	Further arguments passed to or from other methods

Details

The output of 'summary' provides: 1. the goodness-of-fit of the estimated model tested by the likelihood ratio and Pearson's Score Statistics, degrees of freedom and pvalues. Note that degrees of freedom and pvalues are meaningful only for the mph models without inequality constraints; 2. cell-specific statistics: observed and predicted frequencies of the multi-way table, estimated joint probabilities with standard errors, adjusted residuals.

Note

Use 'print' to display only the goodness-of-fit test.

See Also

`mphineq.fit`

Index

∗ **datasets**
accident, 3
depression, 9
drinks, 10
kentucky, 23
madsen, 25
polbirth, 30
relpol, 35
relpolbirth, 36
∗ **generalized marginal interactions**
GMI, 11
∗ **htest**
hmmm.chibar, 14
∗ **logit**
recursive, 33
∗ **marginal models**
GMI, 11
∗ **models**
akaike, 4
create.XMAT, 7
hidden.emfit, 12
hmmm.mlfit, 16
hmmm.model, 18
hmmm.model.X, 21
loglin.model, 24
marg.list, 26
mphineq.fit, 27
∗ **package**
anova.hidden, 5
anova.hmmmfit, 6
getnames, 10
hmmm-package, 2
print.hidden, 31
print.hmmmchibar, 31
print.hmmmfit, 32
summary.hidden, 36
summary.hmmmchibar, 37
summary.hmmmfit, 38
summary.hmmmmod, 39
summary.mphfit, 40

accident, 3
akaike, 4
anova.hidden, 5
anova.hmmmfit, 6, *33*, *39*

create.XMAT, 7, *20*, *22*, *25*, *34*, *39*

depression, 9
drinks, 10

getnames, 10
GMI, 11, *23*

hidden.emfit, *5*, 12, *31*, *37*
hmmm-package, 2
hmmm.chibar, 14, *32*, *38*
hmmm.mlfit, *7, 8*, 16, *20*, *22*, *25*, *29*, *33*, *39*
hmmm.model, *8*, *12*, *14*, *17*, 18, *22, 23*, *25*, *27*, *29*, *34*, *40*
hmmm.model.X, *14*, *17*, *20*, 21, *34*

inv_GMI, *12*, 22

kentucky, 23

loglin.model, 24

madsen, 25
marg.list, *12*, *20*, *22*, 26
mphineq.fit, 27, *41*

polbirth, 30
print.hidden, *5*, *14*, 31, *37*
print.hmmmchibar, *15*, 31, *38*
print.hmmmfit, *7*, *17*, 32, *39*
print.hmmmmod, *20*, *22*
print.hmmmmod *(*summary.hmmmmod*)*, 39
print.mphfit, *29*
print.mphfit *(*summary.mphfit*)*, 40

recursive, *20*, *22*, 33
relpol, 35
relpolbirth, 36

summary.hidden, *5*, *14*, *31*, 36
summary.hmmmchibar, *15*, *32*, 37
summary.hmmmfit, *7, 8*, *17*, *33*, 38
summary.hmmmmod, *20*, *22*, 39
summary.mphfit, *29*, 40