Package ‘cat’

June 3, 2022

Version 0.0-7

Date 2022-04-30

Title Analysis and Imputation of Categorical-Variable Datasets with Missing Values

Author Ported to R by Ted Harding and Fernando Tusell. Original by Joseph L. Schafer <`jls@stat.psu.edu`>.

Maintainer Fernando Tusell <`fernando.tusell@ehu.es`>

Description Performs analysis of categorical-variable with missing values. Implements methods from Schafer, JL, Analysis of Incomplete Multivariate Data, Chapman and Hall.

License file LICENSE

Repository CRAN

Date/Publication 2022-06-03 08:50:10 UTC

NeedsCompilation yes

License_restricts_use no

License_is_FOSS yes

R topics documented:

`belt` *Data on driver injury and seat belt use*

Description

Data on driver injury and seat belt use.

Usage

```
data(belt)
```

Format

The data frame `belt.frame` contains the following columns:

I Injury to driver (I1=Reported by police, I2=Follow up

B Belt use (B1=Reported by police, B2=Follow up

D Damage to vehicle (high, low)

S Sex: Male or Female

Freq Count

Note

A matrix `belt` with similarly named columns exists that can be input directly to functions which do not admit data frames. Both the data frame and matrix include all complete and incomplete cases, from the police reports and follow up study.

Source

Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall, Section 7.4.3, which cites

Hochberg, Y. (1977) On the use of double sampling schemes in analyzing categorical data with misclassification errors, *JASA*, vol. 71, p. 914-921.

bipf *Bayesian Iterative Proportional Fitting (BIPF)*

Description

Markov-Chain Monte Carlo method for simulating posterior draws of cell probabilities under a hierarchical loglinear model

Usage

```
bipf(table,margins, prior=0.5, start, steps=1, showits=FALSE)
```

Arguments

`table`	contingency table (array) to be fitted by a log-linear model. All elements must be non-negative.
`margins`	vector describing the marginal totals to be fitted. A margin is described by the factors not summed over, and margins are separated by zeros. Thus c(1,2,0,2,3,0,1,3) would indicate fitting the (1,2), (2,3), and (1,3) margins in a three-way table, i.e., the model of no three-way association.
`prior`	optional array of hyperparameters specifying a Dirichlet prior distribution. The default is the Jeffreys prior (all hyperparameters = .5). If structural zeros appear in `table`, a prior should be supplied with hyperparameters set to `NA` for those cells.
`start`	starting value for the algorithm. The default is a uniform table. If structural zeros appear in `table`, `start` should contain zeros in those cells and ones elsewhere.
`steps`	number of cycles of Bayesian IPF to be performed.
`showits`	if `TRUE`, reports the iterations so the user can monitor the progress of the algorithm.

Value

array like `table`, but containing simulated cell probabilities that satisfy the loglinear model. If the algorithm has converged, this will be a draw from the actual posterior distribution of the parameters.

Note

The random number generator seed must be set at least once by the function `rngseed` before this function can be used.

The starting value must lie in the interior of the parameter space. Hence, caution should be used when using a maximum likelihood estimate (e.g., from `ipf`) as a starting value. Random zeros in a table may produce mle's with expected cell counts of zero, and any zero in a starting value is interpreted by `bipf` as a structural zero. This difficulty can be overcome by using as a starting value calculated by `ipf` after adding a small positive constant (e.g., 1/2) to each cell.

References

Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall, Chapter 8.

See Also

ipf and rngseed.

Examples

```
data(HairEyeColor)                     # load data
m=c(1,2,0,1,3,0,2,3)                   # no three-way interaction
thetahat <- ipf(HairEyeColor,margins=m,
            showits=TRUE)              # fit model
thetahat <- ipf(HairEyeColor+.5,m)     # find an interior starting value
rngseed(1234567)                       # set random generator seed
theta <- bipf(HairEyeColor,m,
              start=thetahat,prior=0.5,
              steps=50)                # take 50 steps
```

---

crimes *U.S. National Crime Survey*

Description

Victimization status of households on two occasions.

Usage

```
data(crimes)
```

Format

The matrix `crimes` contains the following columns:

V1 Victimization status on first occasion (1=No, 2=Yes)

V1 Victimization status on second occasion (1=No, 2=Yes)

Freq Count

Source

Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall, Section 7.4.3, which cites

Kadane, J.B. (1985) Is victimization chronic? A Bayesian Analysis of multinomial missing data, *Journal of Econometrics*, vol. 29, p. 47-67.

`da.cat` *Data Augmentation algorithm for incomplete categorical data*

Description

Markov-Chain Monte Carlo method for simulating draws from the observed-data posterior distribution of underlying cell probabilities under a saturated multinomial model. May be used in conjunction with `imp.cat` to create proper multiple imputations.

Usage

```
da.cat(s, start, prior=0.5, steps=1, showits=FALSE)
```

Arguments

`s`	summary list of an incomplete categorical dataset created by the function `prelim.cat`.
`start`	starting value of the parameter. This is an array of cell probabilities of dimension `s$d`, such as one created by `em.cat`. If structural zeros appear in the table, starting values for those cells should be zero.
`prior`	optional array of hyperparameters specifying a Dirichlet prior distribution. The default is the Jeffreys prior (all hyperparameters = supplied with hyperparameters set to `NA` for those cells.
`steps`	number of data augmentation steps to be taken. Each step consists of an imputation or I-step followed by a posterior or P-step.
`showits`	if `TRUE`, reports the iterations so the user can monitor the progress of the algorithm.

Details

At each step, the missing data are randomly imputed under their predictive distribution given the observed data and the current value of `theta` (I-step), and then a new value of `theta` is drawn from its Dirichlet posterior distribution given the complete data (P-step). After a suitable number of steps are taken, the resulting value of the parameter may be regarded as a random draw from its observed-data posterior distribution.

When the pattern of observed data is close to a monotone pattern, then `mda.cat` is preferred because it will tend to converge more quickly.

Value

an array like `start` containing simulated cell probabilities.

Note

IMPORTANT: The random number generator seed must be set at least once by the function `rngseed` before this function can be used.

References

Schafer (1996) *Analysis of Incomplete Multivariate Data,* Chapman \& Hall, Chapter 7.

See Also

prelim.cat, rngseed, mda.cat, imp.cat.

Examples

```
data(crimes)
x      <- crimes[,-3]
counts <- crimes[,3]
s <- prelim.cat(x,counts)        # preliminary manipulations
thetahat <- em.cat(s)            # find ML estimate under saturated model
rngseed(7817)                    # set random number generator seed
theta <- da.cat(s,thetahat,50)   # take 50 steps from MLE
ximp  <- imp.cat(s,theta)        # impute once under theta
theta <- da.cat(s,theta,50)      # take another 50 steps
ximp  <- imp.cat(s,theta)        # impute again under new theta
```

---

dabipf *Data augmentation-Bayesian IPF algorithm for incomplete categorical data*

Description

Markov-Chain Monte Carlo method for simulating draws from the observed-data posterior distribution of underlying cell probabilities under hierarchical loglinear models. May be used in conjunction with `imp.cat` to create proper multiple imputations.

Usage

```
dabipf(s, margins, start, steps=1, prior=0.5, showits=FALSE)
```

Arguments

`s`	summary list of an incomplete categorical dataset created by the function `prelim.cat`.
`margins`	vector describing the marginal totals to be fitted. A margin is described by the factors not summed over, and margins are separated by zeros. Thus c(1,2,0,2,3,0,1,3) would indicate fitting the (1,2), (2,3), and (1,3) margins in a three-way table, i.e., the model of no three-way association.
`start`	starting value of the parameter. The starting value should lie in the interior of the parameter space for the given loglinear model. If structural zeros are present, `start` should contain zeros in those positions.
`steps`	number of complete cycles of data augmentation-Bayesian IPF to be performed.

`prior`	optional array of hyperparameters specifying a Dirichlet prior distribution. The default is the Jeffreys prior (all hyperparameters = .5). If structural zeros are present, a prior should be supplied with hyperparameters set to `NA` for those cells.
`showits`	if `TRUE`, reports the iterations so the user can monitor the progress of the algorithm.

Value

array of simulated cell probabilities that satisfy the loglinear model. If the algorithm has converged, this will be a draw from the actual posterior distribution of the parameters.

Note

The random number generator seed must be set at least once by the function `rngseed` before this function can be used.

The starting value must lie in the interior of the parameter space. Hence, caution should be used when using a maximum likelihood estimate (e.g., from `ecm.cat`) as a starting value. Random zeros in a table may produce mle's with expected cell counts of zero. This difficulty can be overcome by using as a starting value a posterior mode calculated by `ecm.cat` with prior hyperparameters greater than one.

References

Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall, Chapter 8.

Examples

```
#
#  Example 1   Based on Schafer's p. 329 and ss. This is a toy version,
#              using a much shorter length of chain than required. To
#              generate results comparable with those in the book, edit
#              the \dontrun{ } line below and comment the previous one.
#
data(belt)
attach(belt.frame)
EB <- ifelse(B1==B2,1,0)
EI <- ifelse(I1==I2,1,0)
belt.frame <- cbind(belt.frame,EB,EI)
colnames(belt.frame)
a <- xtabs(Freq ~ D + S + B2 + I2 + EB + EI,
                data=belt.frame)
m <- list(c(1,2,3,4),c(3,4,5,6),c(1,5),
           c(1,6),c(2,6))
b <- loglin(a,margin=m)                 # fits (DSB2I2)B2I2EBEI)(DEB)(DEI)(SEI)
                                        # in Schafer's p. 304

a <- xtabs(Freq ~ D + S + B2 + I2 + B1 + I1,
                data=belt.frame)
m <- list(c(1,2,5,6),c(1,2,3,4),c(3,4,5,6),
           c(1,3,5),c(1,4,6),c(2,4,6))
```

```
b <- loglin(a,margin=m)                 # fits (DSB1I1)(DSB2I2)(B2I2B1I1)(DB1B2)
                                        #  (DI1I2)(SI1I2) in Schafer's p. 329
s <- prelim.cat(x=belt[,-7],counts=belt[,7])
m <- c(1,2,5,6,0,1,2,3,4,0,3,4,5,6,0,1,3,5,0,1,4,6,0,2,4,6)
theta <- ecm.cat(s,margins=m,           # excruciantingly slow; needs 2558
                   maxits=5000)         # iterations.
rngseed(1234)
#
#   Now ten multiple imputations of the missing variables B2, I2 are
#   generated, by running a chain and taking every 2500th observation.
#   Prior hyperparameter is set at 0.5 as in Shchafer's p. 329
#
imputations <- vector("list",10)

for (i in 1:10) {
cat("Doing imputation ",i,"\n")
  theta <- dabipf(s,m,theta,prior=0.5,   # toy chain; for comparison with
                   steps=25)            # results in Schafer's book the next
                                        # statement should be run,
                                        # rather than this one.
  ## Not run: theta <- dabipf(s,m,theta,prior=0.5,steps=2500)
  imputations[[i]] <- imp.cat(s,theta)
}



detach(belt.frame)
#
#  Example 2   (reproduces analysis performed in Schafer's p. 327.)
#
#  Caveat! I try to reproduce what has been done in that page, but although
#  the general appearance of the boxplots generated below is quite similar to
#  that of  Schafer's Fig. 8.4 (p. 327), the VALUES of the log odds do not
#  quite fall in line with those reported by said author. It doesn't look like
#  the difference can be traced to decimal vs. natural logs. On the other hand,
#  Fig. 8.4 refers to log odds, while the text near the end of page 327 gives
#  1.74 and 1.50 as the means of the *odds* (not log odds). FT, 22.7.2003.
#
#
data(older)                             # reading data
x      <- older[,1:6]                   # preliminary manipulations
counts <- older[,7]
s <- prelim.cat(x,counts)
colnames(x)                             # names of columns
rngseed(1234)
m <- c(1,2,3,4,5,0,1,2,3,5,6,0,4,3)      # model (ASPMG)(ASPMD)(GD) in
                                        # Schafer's p. 327
                                        # do analysis with different priors
theta   <- ecm.cat(s,m,prior=1.5)       # Strong pull to uniform table
                                        # for initial estimates
prob1   <- dabipf(s,m,theta,steps=100,   # Burn-in period
                  prior=0.1)
prob2   <- dabipf(s,m,theta,steps=100,   # Id. with second prior
                  prior=1.5)

lodds   <- matrix(0,5000,2)            # Where to store log odds ratios.

oddsr   <- function(x) {               # Odds ratio of 2 x 2 table.
            o <- (x[1,1]*x[2,2])/
                   (x[1,2]*x[2,1])
            return(o)
            }

for(i in 1:5000) {                     # Now generate 5000 log odds
prob1  <- dabipf(s,m,prob1, prior=0.1)
t1   <- apply(prob1,c(1,2),sum)        # Marginal GD table
                                       # Log odds ratio
lodds[i,1] <- log(oddsr(t1))
prob2  <- dabipf(s,m,prob2, prior=1.5)   # Id. with second prior
t2   <- apply(prob2,c(1,2),sum)
lodds[i,2] <- log(oddsr(t2))
}
lodds  <- as.data.frame(lodds)
colnames(lodds) <- c("0.1","1.5")       # Similar to Schafer's Fig. 8.4.
boxplot(lodds,xlab="Prior hyperparameter")
title(main="Log odds ratio generated with DABIPF (5000 draws)")
summary(lodds)
```

ecm.cat — *ECM algorithm for incomplete categorical data*

Description

Finds ML estimate or posterior mode of cell probabilities under a hierarchical loglinear model

Usage

```
ecm.cat(s, margins, start, prior=1, showits=TRUE, maxits=1000,
eps=0.0001)
```

Arguments

`s`	summary list of an incomplete categorical dataset produced by the function `prelim.cat`.
`margins`	vector describing the sufficient configurations or margins in the desired loglinear model. A margin is described by the factors not summed over, and margins are separated by zeros. Thus c(1,2,0,2,3,0,1,3) would indicate the (1,2), (2,3), and (1,3) margins in a three-way table, i.e., the model of no three-way association. The integers 1,2,... in the specified margins correspond to the columns of the original data matrix `x` that was used to create `s`.

`start`	optional starting value of the parameter. This is an array with dimensions `s$d` whose elements sum to one. The default starting value is a uniform array (equal probabilities in all cells). If structural zeros appear in the table, `start` should contain zeros in those positions and nonzero (e.g. uniform) values elsewhere.
`prior`	optional vector of hyperparameters for a Dirichlet prior distribution. The default is a uniform prior distribution (all hyperparameters = 1) on the cell probabilities, which will result in maximum likelihood estimation. If structural zeros appear in the table, a prior should be supplied with `NA`s in those cells.
`showits`	if `TRUE`, reports the iterations of ECM so the user can monitor the progress of the algorithm.
`maxits`	maximum number of iterations performed. The algorithm will stop if the parameter still has not converged after this many iterations.
`eps`	convergence criterion. This is the largest proportional change in an expected cell count from one iteration to the next. Any expected cell count that drops below 1E-07 times the average cell probability (1/number of non-structural zero cells) is set to zero during the iterations.

Details

At each iteration, performs an E-step followed by a single cycle of iterative proportional fitting.

Value

array of dimension `s$d` containing the ML estimate or posterior mode, assuming that ECM has converged by `maxits` iterations.

Note

If zero cell counts occur in the observed-data tables, the maximum likelihood estimate may not be unique, and the algorithm may converge to different stationary values depending on the starting value. Also, if zero cell counts occur in the observed-data tables, the ML estimate may lie on the boundary of the parameter space. Supplying a prior with hyperparameters greater than one will give a unique posterior mode in the interior of the parameter space. Estimated probabilities for structural zero cells will always be zero.

References

Schafer (1996), *Analysis of Incomplete Multivariate Data.* Chapman \& Hall, Chapter 8

X. L. Meng and D. B. Rubin (1991), "IPF for contingency tables with missing data via the ECM algorithm," Proceedings of the Statistical Computing Section, Amer. Stat. Assoc., 244–247.

See Also

`prelim.cat`, `em.cat`, `logpost.cat`

Examples

```
data(older)                          # load data
#
#  Example 1
#
older[1:2,]                          # see partial content; older.frame also
                                     # available.
s <- prelim.cat(older[,-7],older[,7]) # preliminary manipulations
m <- c(1,2,5,6,0,3,4)                # margins for restricted model
try(thetahat1 <- ecm.cat(s,margins=m))# will complain
thetahat2 <- ecm.cat(s,margins=m,prior=1.1)
                                     # same model with prior information
logpost.cat(s,thetahat2)             # loglikelihood under thetahat2
#
#  Example 2   (reproduces analysis performed in Schafer's p. 327.)
#
m1 <- c(1,2,3,5,6,0,1,2,4,5,6,0,3,4)  # model (ASPMG)(ASPMD)(GD) in
                                     # Schafer's p. 327
theta1 <- ecm.cat(s,margins=m1,
                       prior=1.1)     # Prior to bring MLE away from boundary.
m2     <- c(1,2,3,5,6,0,1,2,4,5,6)    # model (ASPMG)(ASPMD)
theta2 <- ecm.cat(s,margins=m2,
                       prior=1.1)
lik1   <- logpost.cat(s,theta1)       # posterior log likelihood.
lik2   <- logpost.cat(s,theta2)       # id. for restricted model.
lrt    <- -2*(lik2-lik1)              # for testing significance of (GD)
p      <- 1 - pchisq(lrt,1)           # significance level
cat("LRT statistic for \n(ASMPG)(ASMPD) vs. (ASMPG)(ASMPD)(GD): ",lrt," with p-value = ",p)
```

`em.cat` *EM algorithm for incomplete categorical data*

Description

Finds ML estimate or posterior mode of cell probabilities under the saturated multinomial model.

Usage

```
em.cat(s, start, prior=1, showits=TRUE, maxits=1000,
eps=0.0001)
```

Arguments

`s`	summary list of an incomplete categorical dataset produced by the function `prelim.cat`.
`start`	optional starting value of the parameter. This is an array with dimensions `s$d` whose elements sum to one. The default starting value is a uniform array (equal probabilities in all cells). If structural zeros appear in the table, `start` should contain zeros in those positions and nonzero (e.g. uniform) values elsewhere.

`prior`	optional vector of hyperparameters for a Dirichlet prior distribution. The default is a uniform prior distribution (all hyperparameters = 1) on the cell probabilities, which will result in maximum likelihood estimation. If structural zeros appear in the table, a prior should be supplied with `NA`s in those cells.
`showits`	if `TRUE`, reports the iterations of EM so the user can monitor the progress of the algorithm.
`maxits`	maximum number of iterations performed. The algorithm will stop if the parameter still has not converged after this many iterations.
`eps`	convergence criterion. This is the largest proportional change in an expected cell count from one iteration to the next. Any expected cell count that drops below 1E-07 times the average cell probability (1/number of non-structural zero cells) is set to zero during the iterations.

Value

array of dimension `s$d` containing the ML estimate or posterior mode, assuming that EM has converged by `maxits` iterations.

Note

If zero cell counts occur in the observed-data table, the maximum likelihood estimate may not be unique, and the algorithm may converge to different stationary values depending on the starting value. Also, if zero cell counts occur in the observed-data table, the ML estimate may lie on the boundary of the parameter space. Supplying a prior with hyperparameters greater than one will give a unique posterior mode in the interior of the parameter space. Estimated probabilities for structural zero cells will always be zero.

References

Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall, Section 7.3.

See Also

`prelim.cat`, `ecm.cat`, `logpost.cat`

Examples

```
data(crimes)
crimes
s <- prelim.cat(crimes[,1:2],crimes[,3])     # preliminary manipulations
thetahat <- em.cat(s)                        # mle under saturated model
logpost.cat(s,thetahat)                      # loglikelihood at thetahat
```

`imp.cat` *Impute missing categorical data*

Description

Performs single random imputation of missing values in a categorical dataset under a user-supplied value of the underlying cell probabilities.

Usage

```
imp.cat(s, theta)
```

Arguments

`s`	summary list of an incomplete categorical dataset created by the function `prelim.cat`.
`theta`	parameter value under which the missing data are to be imputed. This is an array of cell probabilities of dimension `s$d` whose elements sum to one, such as produced by `em.cat`, `ecm.cat`, `da.cat`, `mda.cat` or `dabipf`.

Details

Missing data are drawn independently for each observational unit from their conditional predictive distribution given the observed data and `theta`.

Value

If the original incomplete dataset was in ungrouped format (`s$grouped=F`), then a matrix like `s$x` except that all `NA`s have been filled in.

If the original dataset was grouped, then a list with the following components:

`x`	Matrix of levels for categorical variables
`counts`	vector of length `nrow(x)` containing frequencies or counts corresponding to the levels in `x`.

Note

IMPORTANT: The random number generator seed must be set by the function `rngseed` at least once in the current session before this function can be used.

See Also

`prelim.cat`, `rngseed`, `em.cat`, `da.cat`, `mda.cat`, `ecm.cat`, `dabipf`

Examples

```
data(crimes)
x      <- crimes[,-3]
counts <- crimes[,3]
s <- prelim.cat(x,counts)        # preliminary manipulations
thetahat <- em.cat(s)            # find ML estimate under saturated model
rngseed(7817)                    # set random number generator seed
theta <- da.cat(s,thetahat,50)   # take 50 steps from MLE
ximp  <- imp.cat(s,theta)        # impute once under theta
theta <- da.cat(s,theta,50)      # take another 50 steps
ximp  <- imp.cat(s,theta)        # impute again under new theta
```

`ipf` *Iterative Proportional Fitting*

Description

ML estimation for hierarchical loglinear models via conventional iterative proportional fitting (IPF).

Usage

```
ipf(table, margins, start, eps=0.0001, maxits=50, showits=TRUE)
```

Arguments

`table`	contingency table (array) to be fit by a log-linear model. All elements must be non-negative.
`margins`	vector describing the marginal totals to be fitted. A margin is described by the factors not summed over, and margins are separated by zeros. Thus c(1,2,0,2,3,0,1,3) would indicate fitting the (1,2), (2,3), and (1,3) margins in a three-way table, i.e., the model of no three-way association.
`start`	starting value for IPF algorithm. The default is a uniform table. If structural zeros appear in `table`, `start` should contain zeros in those cells and ones elsewhere.
`eps`	convergence criterion. This is the largest proportional change in an expected cell count from one iteration to the next. Any expected cell count that drops below 1E-07 times the average cell probability (1/number of non-structural zero cells) is set to zero during the iterations.
`maxits`	maximum number of iterations performed. The algorithm will stop if the parameter still has not converged after this many iterations.
`showits`	if `TRUE`, reports the iterations of IPF so the user can monitor the progress of the algorithm.

Value

array like `table`, but containing fitted values (expected frequencies) under the loglinear model.

DETAILS

This function is usually used to compute ML estimates for a loglinear model. For ML estimates, the array `table` should contain the observed frequencies from a cross-classified contingency table. Because this is the "cell-means" version of IPF, the resulting fitted values will add up to equals `sum(table)`. To obtain estimated cell probabilities, rescale the fitted values to sum to one.

This function may also be used to compute the posterior mode of the multinomial cell probabilities under a Dirichlet prior. For a posterior mode, set the elements of `table` to (observed frequencies + Dirichlet hyperparameters - 1). Then, after running IPF, rescale the fitted values to sum to one.

Note

This function is essentially the same as the old S function `loglin`, but results are computed to double precision. See `help(loglin)` for more details.

References

Agresti, A. (1990) Categorical Data Analysis. J. Wiley & Sons, New York.

Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall, Chapter 8.

See Also

`ecm.cat`, `bipf`

Examples

```
data(HairEyeColor)                     # load data
m=c(1,2,0,1,3,0,2,3)                   # no three-way interaction
fit1 <- ipf(HairEyeColor,margins=m,
            showits=TRUE)              # fit model
X2 <- sum((HairEyeColor-fit1)^2/fit1)  # Pearson chi square statistic
df <- prod(dim(HairEyeColor)-1)        # Degrees of freedom for this example
1 - pchisq(X2,df)                      # p-value for fit1
```

`logpost.cat` *Log-posterior density for incomplete categorical data*

Description

Calculates the observed-data loglikelihood or log-posterior density for incomplete categorical data under a specified value of the underlying cell probabilities, e.g. as resulting from em.cat or ecm.cat.

Usage

```
logpost.cat(s, theta, prior)
```

Arguments

`s`	summary list of an incomplete categorical dataset created by the function `prelim.cat`.
`theta`	an array of cell probabilities of dimension `s$d`
`prior`	optional vector of hyperparameters for a Dirichlet prior distribution. The default is a uniform prior distribution (all hyperparameters = 1) on the cell probabilities, which will result in evaluation of the loglikelihood. If structural zeros appear in the table, a prior should be supplied with NAs in those cells and ones (or other hyperparameters) elsewhere.

Details

This is the loglikelihood or log-posterior density that ignores the missing-data mechanism.

Value

the value of the observed-data loglikelihood or log-posterior density function at `theta`

References

Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall. Section 7.3.

See Also

`prelim.cat`, `em.cat`, `ecm.cat`

Examples

```
data(older)                             # load data
older[1:2,c(1:4,7)]                     # see partial content; older.frame also
                                        # available.
s <- prelim.cat(older[,1:4],older[,7]) # preliminary manipulations
m <- c(1,2,0,3,4)                       # margins for restricted model
thetahat1 <- ecm.cat(s,margins=m)       # mle
logpost.cat(s,thetahat1)                # loglikelihood at thetahat1
```

`mda.cat` *Monotone Data Augmentation algorithm for incomplete categorical data*

Description

Markov-Chain Monte Carlo method for simulating draws from the observed-data posterior distribution of underlying cell probabilities under a saturated multinomial model. May be used in conjunction with `imp.cat` to create proper multiple imputations. Tends to converge more quickly than `da.cat` when the pattern of observed data is nearly monotone.

Usage

```
mda.cat(s, start, steps=1, prior=0.5, showits=FALSE)
```

Arguments

`s`	summary list of an incomplete categorical dataset created by the function `prelim.cat`.
`start`	starting value of the parameter. This is an array of cell probabilities of dimension `s$d`, such as one created by `em.cat`. If structural zeros appear in the table, starting values for those cells should be zero.
`steps`	number of data augmentation steps to be taken. Each step consists of an imputation or I-step followed by a posterior or P-step.
`prior`	optional vector of hyperparameters specifying a Dirichlet prior distribution. The default is the Jeffreys prior (all hyperparameters = supplied with hyperparameters set to `NA` for those cells.
`showits`	if `TRUE`, reports the iterations so the user can monitor the progress of the algorithm.

Details

At each step, the missing data are randomly imputed under their predictive distribution given the observed data and the current value of `theta` (I-step) Unlike `da.cat`, however, not all of the missing data are filled in, but only enough to complete a monotone pattern. Then a new value of `theta` is drawn from its Dirichlet posterior distribution given the monotone data (P-step). After a suitable number of steps are taken, the resulting value of the parameter may be regarded as a random draw from its observed-data posterior distribution.

For good performance, the variables in the original data matrix `x` (which is used to create `s`) should be ordered according to their rates of missingness from most observed (in the first columns) to least observed (in the last columns).

Value

an array like `start` containing simulated cell probabilities.

Note

IMPORTANT: The random number generator seed must be set at least once by the function `rngseed` before this function can be used.

References

Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall, Chapter 7.

See Also

`prelim.cat`, `rngseed`, `da.cat`, `imp.cat`.

Examples

```
data(older)
x      <- older[1:80,1:4]               # subset of the data with
counts <- older[1:80,7]                 # monotone pattern.
s <- prelim.cat(x,counts)               # preliminary manipulations
thetahat <- em.cat(s)                   # mle under saturated model
rngseed(7817)                           # set random generator seed
theta <- mda.cat(s,thetahat,50)         # take 50 steps from mle
ximp <- imp.cat(s,theta)                # impute under theta
theta <- mda.cat(s,theta,50)            # take another 50 steps
ximp <- imp.cat(s,theta)                # impute under new theta
```

---

mi.inference *Multiple imputation inference*

Description

Combines estimates and standard errors from m complete-data analyses performed on m imputed datasets to produce a single inference. Uses the technique described by Rubin (1987) for multiple imputation inference for a scalar estimand.

Usage

```
mi.inference(est, std.err, confidence=0.95)
```

Arguments

`est`	a list of \$m\$ (at least 2) vectors representing estimates (e.g., vectors of estimated regression coefficients) from complete-data analyses performed on \$m\$ imputed datasets.
`std.err`	a list of \$m\$ vectors containing standard errors from the complete-data analyses corresponding to the estimates in `est`.
`confidence`	desired coverage of interval estimates.

Value

a list with the following components, each of which is a vector of the same length as the components of `est` and `std.err`:

`est`	the average of the complete-data estimates.
`std.err`	standard errors incorporating both the between and the within-imputation uncertainty (the square root of the "total variance").
`df`	degrees of freedom associated with the t reference distribution used for interval estimates.
`signif`	P-values for the two-tailed hypothesis tests that the estimated quantities are equal to zero.

`lower`	lower limits of the (100*confidence)% interval estimates.
`upper`	upper limits of the (100*confidence)% interval estimates.
`r`	estimated relative increases in variance due to nonresponse.
`fminf`	estimated fractions of missing information.

METHOD

Uses the method described on pp. 76-77 of Rubin (1987) for combining the complete-data estimates from $m$ imputed datasets for a scalar estimand. Significance levels and interval estimates are approximately valid for each one-dimensional estimand, not for all of them jointly.

References

Fienberg, S.E. (1981) *The Analysis of Cross-Classified Categorical Data*, MIT Press, Cambridge.

Rubin (1987) *Multiple Imputation for Nonresponse in Surveys,* Wiley, New York,

Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall, Chapter 8.

See Also

`dabipf`, `imp.cat`

Examples

```
#
#  Example 1   Based on Schafer's p. 329 and ss. This is a toy version,
#              using a much shorter length of chain than required. To
#              generate results comparable with those in the book, edit
#              the \dontrun{ } line below and comment the previous one.
#
data(belt)
attach(belt.frame)

oddsr   <- function(x) {                # Odds ratio of 2 x 2 table.
            o <- (x[1,1]*x[2,2])/
                   (x[1,2]*x[2,1])
            o.sd <- sqrt(1/x[1,1] +     # large sample S.D. (Fienberg,
                      1/x[1,2] +        # p. 18)
                      1/x[2,1] +
                      1/x[2,2])
            return(list(o=o,sd=o.sd))
            }

colns <- colnames(belt.frame)

a <- xtabs(Freq ~ D + S + B2 + I2 + B1 + I1,
                data=belt.frame)
m <- list(c(1,2,5,6),c(1,2,3,4),c(3,4,5,6),
          c(1,3,5),c(1,4,6),c(2,4,6))
b <- loglin(a,margin=m)                 # fits (DSB1I1)(DSB2I2)(B2I2B1I1)(DB1B2)
                                        #  (DI1I2)(SI1I2) in Schafer's p. 329
```

```
s <- prelim.cat(x=belt[,-7],counts=belt[,7])
m <- c(1,2,5,6,0,1,2,3,4,0,3,4,5,6,0,1,3,5,0,1,4,6,0,2,4,6)
theta <- ecm.cat(s,margins=m,            # excruciantingly slow; needs 2558
                 maxits=5000)            # iterations.
rngseed(1234)
#
#   Now ten multiple imputations of the missing variables B2, I2 are
#   generated, by running a chain and taking every 2500th observation.
#   Prior hyperparameter is set at 0.5 as in Schafer's p. 329
#
est <- std.error <- vector("list",10)

for (i in 1:10) {
cat("Doing imputation ",i,"\n")
  theta <- dabipf(s,m,theta,prior=0.5,   # toy chain; for comparison with
                 steps=25)               # results in Schafer's book the next
                                         # statement should be run,
                                         # rather than this one.
  ## Not run: theta <- dabipf(s,m,theta,prior=0.5,steps=2500)
  imp<- imp.cat(s,theta)
  imp.frame <- cbind(imp$x,imp$counts)
  colnames(imp.frame) <- colns
  a <- xtabs(Freq ~ B2 + I2,             # 2 x 2 table relating belt use
                  data=imp.frame)        # and injury
  print(a)
  odds <- oddsr(a)                       # odds ratio and std.dev.
  est[[i]] <- odds$o - 1                 # check deviations from 1 of
  std.error[[i]] <- odds$sd              # odds ratio
}
odds <- mi.inference(est,std.error)
print(odds)
detach(belt.frame)
```

---

older *Older people dataset*

Description

Data from the Protective Services Project for Older Persons

Usage

```
data(older)
```

Format

The data frame `older.frame` contains the following columns:

M Mental status

P ysical status

D Survival status (deceased or not)

G Group membership: E=experimental, C=control)

A Age: Under75 and 75+

S Sex: Male or Female

Freq Count

Note

A matrix `older` with similarley named columns exists that can be input directly to functions which do not admit data frames.

Source

Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall, Section 7.3.5.

`prelim.cat` *Preliminary manipulations on incomplete categorical data*

Description

This function performs grouping and sorting operations on categorical datasets with missing values. It creates a list that is needed for input to em.cat, da.cat, imp.cat, etc.

Usage

```
prelim.cat(x, counts, levs)
```

Arguments

`x`	categorical data matrix containing missing values. The data may be provided either in ungrouped or grouped format. In ungrouped format, the rows of x correspond to individual observational units, so that nrow(x) is the total sample size. In grouped format, the rows of x correspond to distinct covariate patterns; the frequencies are provided through the `counts` argument. In either format, the columns correspond to variables. The categories must be coded as consecutive positive integers beginning with 1 (1,2,. . .), and missing values are denoted by `NA`.
`counts`	optional vector of length `nrow(x)` giving the frequencies corresponding to the covariate patterns in x. The total sample size is `sum(counts)`. If `counts` is missing, the data are assumed to be ungrouped; this is equivalent to taking `counts` equal to `rep(1,nrow(x))`.
`levs`	optional vector of length `ncol(x)` indicating the number of levels for each categorical variable. If missing, `levs[j]` is taken to be `max(x[,j],na.rm=T)`.

Value

a list of seventeen components that summarize various features of x after the data have been sorted by missingness patterns and grouped according to the observed values. Components that might be of interest to the user include:

`nmis`	a vector of length `ncol(x)` containing the number of missing values for each variable in x.
`r`	matrix of response indicators showing the missing data patterns in x. Dimension is (m,p) where m is number of distinct missingness patterns in the rows of x, and p is the number of columns in x. Observed values are indicated by 1 and missing values by 0. The row names give the number of observations in each pattern, and the columns correspond to the columns of x.
`d`	vector of length `ncol(x)` indicating the number of levels for each variable. The complete-data contingency table would be an array with these dimensions. Identical to `levs` if `levs` was supplied.
`ncells`	number of cells in the cross-classified contingency table, equal to `prod(d)`.

References

Chapters 7–8 of Schafer (1996) *Analysis of Incomplete Multivariate Data.* Chapman \& Hall.

See Also

`em.cat`, `ecm.cat`, `da.cat`,`mda.cat`, `dabipf`, `imp.cat`

Examples

```
data(crimes)
crimes
s <- prelim.cat(crimes[,1:2],crimes[,3])   # preliminary manipulations
s$nmis                        # see number of missing observations per variable
s$r                           # look at missing data patterns
```

`rngseed` *Initialize random number generator seed*

Description

Seeds the random number generator

Usage

```
rngseed(seed)
```

Arguments

`seed`	a positive number, preferably a large integer.

Value

`NULL`.

Note

The random number generator seed must be set at least once by this function before the simulation or imputation functions in this package (`da.cat`, `imp.cat`, etc.) can be used.

Index