Package ‘ACD’

February 19, 2015

Version 1.5.3

Date 2013-30-10

Title Categorical data analysis with complete or missing responses

Author Frederico Zanqueta Poleto, Julio da Mota Singer, Carlos Daniel Paulino, Fabio Mathias Correa and Enio Galinkin Jelihovschi

Maintainer Fabio Mathias Correa <`fmcorrea@uesc.br`>

Description Categorical data analysis with complete or missing responses

Depends R(>= 2.13.0)

License GPL (>= 2)

Repository CRAN

Date/Publication 2013-10-31 20:59:03

NeedsCompilation no

R topics documented:

`funlinWLS` *Fitting Functional Linear Models via Weighted Least Squares*

Description

`funlinWLS` fits functional linear models by WLS (weighted least squares). For complete data, it is based on a object of the class `readCatdata`. For missing data, it is based on a object of the class `satMarML` (under MAR or MCAR) or `satMcarWLS` (under MCAR). For both complete and missing data, another alternative is to use as inputs the maximum likelihood (ML) or any other best asymptotically normal (BAN) estimate for the product-multinomial probabilities under a fitted model (which may encompass MAR, MCAR or MNAR assumption) and a consistent estimate of its asymptotic covariance matrix. Depending on the formulation (freedom equations or constraints) and on the model specification, different arguments must be informed.

Usage

```
funlinWLS(model, obj, theta, Vtheta, A1, A2, A3, A4, A5, A6, A7, A8,
A9,X, U, XL, UL, zeroN, PI1, PI2, PI3, PI4, PI5, PI6, PI7, PI8, PI9)
```

Arguments

`model`	vector of functions to be modeled linearly; they need to be specified in the order with which they are applied to the vector of proportions; the supported functions are: linear (`"lin"`), logarithmic (`"log"`), exponential (`"exp"`), and addition of constants (`"add"`).
`obj`	an object of class `readCatdata`, `satMarML` or `satMcarWLS`.
`theta`	BAN estimate of the product-multinomial probabilities.
`Vtheta`	consistent estimate of asymptotic covariance matrix of the estimators of theta.
`A1`	a matrix for the 1st linear function; for linear model (`model = "lin"`) the default is a matrix diag(*S*) %x% cbind(diag(*R-1*),rep(*0*,*R-1*)) which discards the last element of the vector of probabilies of each multinomial; for log-linear model (`model=c("lin", "log")`) the default is a matrix diag(*S*) %x% cbind(diag(*R-1*), rep(*-1*,*R-1*)) which generates logits with the last category as the baseline.
`A2`	a matrix for the 2nd linear function.
`A3`	a matrix for the 3rd linear function.
`A4`	a matrix for the 4th linear function.
`A5`	a matrix for the 5th linear function.
`A6`	a matrix for the 6th linear function.
`A7`	a matrix for the 7th linear function.
`A8`	a matrix for the 8th linear function.
`A9`	a matrix for the 9th linear function.
`X`	a model specification matrix for freedom equation formulation for all functional linear models; for log-linear model (`model=c("lin","log")`), this is used for the ordinary log-linear specification.

`U`	a matrix for constraint formulation for all functional linear models; for log-linear model (`model=c("lin","log")`), this is used for the ordinary log-linear specification.
`XL`	a model specification matrix for freedom equation formulation of generalized log-linear models (`model=c("lin","log")`).
`UL`	a matrix for constraint formulation of generalized log-linear models (`model=c("lin", "log")`).
`zeroN`	for complete data, the vector has S values that are used to replace null frequencies in each subpopulation in order to compute the proportions used in the estimate for the covariance matrix; the default value is `1/(R*ns)`, where *ns* is the sample size associated to the corresponding subpopulation; for missing data, this is not required as the possible replacements occur either at `satMarML` or at `satMcarWLS`.
`PI1`	the 1st vector of constants to be added.
`PI2`	the 2nd vector of constants to be added.
`PI3`	the 3rd vector of constants to be added.
`PI4`	the 4th vector of constants to be added.
`PI5`	the 5th vector of constants to be added.
`PI6`	the 6th vector of constants to be added.
`PI7`	the 7th vector of constants to be added.
`PI8`	the 8th vector of constants to be added.
`PI9`	the 9th vector of constants to be added.

Details

Every linear function demands the specification of a matrix in Ai, where i may vary from *1* to *9*; such matrices must be numbered from right to left, in the order of which the operations are applied. Similarly, the user needs to specify a vector *PIi* for every addition of constants.

Examples of functions (for simplicity, consider "*" as a matrix multiplication in the following functions)

Function F(Theta)	*model*
A1*Theta	*"lin"*
log(Theta)	*"log"*
exp(Theta)	*"exp"*
PI1+Theta	*"add"*
A1*log(Theta)	*c("lin","log")*
exp[A1*log(Theta)]	*c("exp","lin","log")*
PI3+exp[PI2+A1*log(PI1+Theta)]	*c("add","exp","add","lin","log","add")*
PI1+exp(A4*logA3*exp[A2*log(A1*Theta)])	*c("add","exp","lin","log","lin","exp","lin","log","lin")*

Function F(Theta)	Arguments that must be supplied
A1*Theta	A1

log(Theta)	(none)
exp(Theta)	(none)
PI1+Theta	PI1
A1*log(Theta)	A1
exp[A1*log(Theta)]	A1
PI3+exp[PI2+A1*log(PI1+Theta)]	A1, PI1, PI2, PI3
PI1+exp(A4*logA3*exp[A2*log(A1*Theta)])	A1, A2, A3, A4, PI1

Functional linear models may be fitted to the functions F(Theta) using a freedom equation formulation F(Theta)=X%*%Beta, where the elements of Beta are the parameters to be estimated, or using a constraint formulation U%*%F(Theta)=0. Both formulations lead to an equivalent model fit if U%*%X=0.

Specifically for log-linear models (`model=c("lin","log")`), X and U are used for ordinary log-linear models, and XL and UL are used for generalized log-linear models, namely log(Theta) = nu+X%*%Beta, U%*%log(Theta) = 0, A1%*%log(Theta) = XL%*%Beta, UL%*%A1%*%log(Theta) = 0, where nu are non-estimated parameters included only to satisfy the natural constraints of the product-multinomial distribution.

The generic functions `print` and `summary` are used to print the results and to obtain a summary thereof.

Value

An object of the class `funlinWLS` is a list containing most of the components of the argument `obj` as well as the following components:

`thetaH`	vector of WLS estimates for all product-multinomial probabilities under the functional linear model (in the case of missing data, conditional on the previously assumed model for the missingness mechanism.
`VthetaH`	corresponding estimated covariance matrix.
`beta`	vector of WLS estimates for parameters of the functional linear model (only for the freedom equation formulation).
`Vbeta`	corresponding estimated covariance matrix (only for the freedom equation formulation).
`Fu`	observed functions, without model constraints.
`VFu`	corresponding estimated covariance matrix.
`FH`	WLS estimates for the functions under the fitted model.
`VFH`	corresponding estimated covariance matrix.
`QwH`	Wald statistic for testing the goodness of fit of the functional linear model (for missing data, conditional on the assumed missingness mechanism).
`glH`	degrees of freedom for testing the goodness of fit of the functional linear model (for missing data, conditional on the assumed missingness mechanism).
`ystH`	for complete data, it contains the WLS estimates for the frequencies under the functional linear model; for missing data, it contains the WLS estimates for the

augmented frequencies under both the linear model and the assumed missingness mechanism; for both missing and complete data, this is computed only for linear and certain log-linear models wherein it is possible to estimate the marginal probabilities of categorization.

Author(s)

Frederico Zanqueta Poleto(`<frederico@poleto.com>`)
Julio da Motta Singer (`<jmsinger@ime.usp.br>`)
Carlos Daniel Paulino (`<daniel.paulino@math.ist.utl.pt>`)
with the collaboration of
Fabio Mathias Correa (`<fmcorrea@uesc.br>`)
Enio Galinkin Jelihovschi (`<eniojelihovs@gmail.com>`)

References

Paulino, C.D. e Singer, J.M. (2006). *Analise de dados categorizados* (in Portuguese). Sao Paulo: Edgard Blucher.

Poleto, F.Z. (2006). *Analise de dados categorizados com omissao* (in Portuguese). Dissertacao de mestrado. IME-USP. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2007). *Analyzing categorical data with complete or missing responses using the Catdata package*. Unpublished vignette. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2012). A product-multinomial framework for categorical data analysis with missing responses. To appear in *Brazilian Journal of Probability and Statistics*. `http://imstat.org/bjps/papers/BJPS198.pdf`.

Singer, J. M., Poleto, F. Z. and Paulino, C. D. (2007). Catdata: software for analysis of categorical data with complete or missing responses. *Actas de la XII Reunion Cientifica del Grupo Argentino de Biometria y I Encuentro Argentino-Chileno de Biometria*. `http://www.poleto.com/SingerPoletoPaulino2007GAB.pdf`.

Examples

```
#Example 11.2 of Paulino and Singer (2006)
e112.TF<-c(192,1,5,2,146,5,11,12,71)

e112.catdata<-readCatdata(TF=e112.TF)

e112.U<-rbind(c(0,-1, 0,1,0, 0,0,0),
               c(0, 0,-1,0,0, 0,1,0),
               c(0, 0, 0,0,0,-1,0,1))

e112.X<-rbind(c(1,0,0,0,0),
               c(0,1,0,0,0),
               c(0,0,1,0,0),
               c(0,1,0,0,0),
              c(0,0,0,1,0),
               c(0,0,0,0,1),
```

```
               c(0,0,1,0,0),
               c(0,0,0,0,1))

#Two equivalent ways of fitting the same symmetry model
e112.linwls1<-funlinWLS(model="lin",obj=e112.catdata,U=e112.U)
e112.linwls2<-funlinWLS(model="lin",obj=e112.catdata,X=e112.X)
e112.linwls1 #constraint formulation
e112.linwls2 #freedom equation formulation
summary(e112.linwls1)

#Example 11.5 of Paulino and Singer (2006)
e115.TF<-c(3,25,32,68)
e115.catdata<-readCatdata(TF=e115.TF)
e115.U<-c(1,-1,-1,1)

e115.X<-rbind(c(0,0),c(0,1),c(1,0),c(1,1))

e115.X2<-rbind(c(0,0,0),c(0,1,0),c(1,0,0),c(1,1,1))

e115.loglinwls1<-funlinWLS(model=c("lin", "log"), obj=e115.catdata,
U=e115.U)
e115.loglinwls2<-funlinWLS(model=c("lin", "log"), obj=e115.catdata,
X=e115.X)
e115.loglinwls3<-funlinWLS(model=c("lin", "log"), obj=e115.catdata,
X=e115.X2)
e115.loglinwls4<-funlinWLS(model=c("lin", "log"), obj=e115.catdata,
A1=c(1,-1,-1,1), XL=1)

#Independence ordinary log-linear model, constraint formulation
e115.loglinwls1

#Independence ordinary log-linear model, freedom equation formulation
e115.loglinwls2

#Saturated ordinary log-linear model, freedom equation formulation
e115.loglinwls3

#Saturated generalized log-linear model, freedom equation formulation
e115.loglinwls4

#95% confidence interval for log-odds ratio and for odds ratio

round(e115.loglinwls4$beta+c(-1,1)*qnorm(0.975)*sqrt(e115.loglinwls4$Vbeta),3)
round(exp(e115.loglinwls4$beta),3)
round(exp(e115.loglinwls4$beta+c(-1,1)*qnorm(0.975)*sqrt(e115.loglinwls4$Vbeta)),3)

#Example 11.3 of Paulino and Singer (2006)
e113.TF<-c(11,5,0,14,34,7,2,13,11)
e113.catdata<-readCatdata(TF=e113.TF)

e113.U<-rbind(c(0, 1,1,-1,0,0,-1, 0),
              c(0,-1,0, 1,0,1, 0,-1))
```

```
e113.X<-rbind(c(1, 0, 0,0,0,0),
               c(0, 1, 0,0,0,0),
               c(0,-1, 1,0,1,0),
               c(0, 0, 1,0,0,0),
              c(0, 0, 0,1,0,0),
               c(0, 1,-1,0,0,1),
               c(0, 0, 0,0,1,0),
               c(0, 0, 0,0,0,1))

e113.linwls1<-funlinWLS(model=”lin”,obj=e113.catdata,U=e113.U)
e113.linwls2<-funlinWLS(model=”lin”,obj=e113.catdata,X=e113.X)

e113.A<-rbind(c(1,1,1,0,0,0,0,0,0),
               c(0,0,0,1,1,1,0,0,0),
               c(1,0,0,1,0,0,1,0,0),
               c(0,1,0,0,1,0,0,1,0))

e113.U2<-rbind(c(1,0,-1, 0),c(0,1, 0,-1))
e113.X2<-rbind(c(1,0),c(0,1),c(1,0),c(0,1))

e113.linwls3<-funlinWLS(model=”lin”,obj=e113.catdata,A1=e113.A,U=e113.U2)
e113.linwls4<-funlinWLS(model=”lin”,obj=e113.catdata,A1=e113.A,X=e113.X2)

#Four equivalent ways of fitting the same marginal homogeneity model
e113.linwls1
e113.linwls2
e113.linwls3
e113.linwls4

#Example 11.12 of Paulino and Singer (2006)
e1112.TF<-c(11,5,0,14,34,7,2,13,11)
e1112.catdata<-readCatdata(TF=e1112.TF)

e1112.A1<-rbind(c(rep(c(1,0,0,0),2),1),rep(1,9),
  kronecker(diag(3),t(rep(1,3))),kronecker(t(rep(1,3)),diag(3)))

e1112.A2<-rbind(cbind(diag(2),matrix(0,2,6)),
cbind(matrix(0,3,2),kronecker(t(rep(1,2)),diag(3))))

e1112.A3<-cbind(c(1,0),c(1,1),
tcrossprod(-c(2,1),(rep(1,3))))

e1112.A4<-t(c(1,-1))

e1112.kappa<-funlinWLS(model = c(”add”, ”exp”, ”lin”, ”log”, ”lin”,
”exp”, ”lin”, ”log”, ”lin”),
obj=e1112.catdata, A1=e1112.A1, A2=e1112.A2, A3=e1112.A3, A4=e1112.A4,
PI1=-1, X=1)

# confidence interval
round(e1112.kappa$beta+c(-1,1)*qnorm(0.975)*sqrt(e1112.kappa$Vbeta),3)

#weighted kappa (Spitzer, Cohen, Fleiss e Endicott, 1967)
```

```
  #squared weights (Fleiss e Cohen, 1973)
W1<-c(1,0.75,0,0.75,1,0.75,0,0.75,1)

#absolute weights (Cicchetti e Allison, 1971)
W2<-c(1,0.5,0,0.5,1,0.5,0,0.5,1)

e1112.w1A1<-rbind(t(W1),rep(1,9),kronecker(diag(3),t(rep(1,3))),
  kronecker(t(rep(1,3)),diag(3)))

e1112.w2A1<-rbind(t(W2),rep(1,9),kronecker(diag(3),t(rep(1,3))),
  kronecker(t(rep(1,3)),diag(3)))

e1112.wA2<-rbind(cbind(diag(2),matrix(0,2,6)),cbind(matrix(0,9,2),
 cbind(kronecker(diag(3),rep(1,3)),kronecker(rep(1,3),diag(3)))))

e1112.w1A3<-cbind(c(1,0),c(1,1),kronecker(-c(2,1),t(W1)))

e1112.w2A3<-cbind(c(1,0),c(1,1),kronecker(-c(2,1),t(W2)))

e1112.kappaw1<-funlinWLS(model=c("add", "exp", "lin", "log", "lin",
"exp", "lin", "log", "lin"),
 obj=e1112.catdata, A1=e1112.w1A1, A2=e1112.wA2, A3=e1112.w1A3, A4=e1112.A4,
 PI1=-1, X=1)

e1112.kappaw2<-funlinWLS(model=c("add", "exp", "lin", "log", "lin",
      "exp", "lin","log", "lin"),
 obj=e1112.catdata, A1=e1112.w2A1, A2=e1112.wA2, A3=e1112.w2A3, A4=e1112.A4,
 PI1=-1, X=1)

#Example 1 of Poleto et al (2012)
smoking.TF<-rbind(c(167,17,19,10,1,3,52,10,11, 176,24,121, 28,10,12),
                  c(120,22,19, 8,5,1,39,12,12, 103, 3, 80, 31, 8,14))

smoking.Zp<-kronecker(t(rep(1,2)),cbind(kronecker(diag(3),rep(1,3)),
kronecker(rep(1,3),diag(3))))

smoking.Rp<-rbind(c(3,3),c(3,3))

smoking.catdata<-readCatdata(TF=smoking.TF,Zp=smoking.Zp,Rp=smoking.Rp)

smoking.catdata #Proportions of the complete data
smoking.satmarml<-satMarML(smoking.catdata)
smoking.satmcarml<-satMarML(smoking.catdata,missing="MCAR")
smoking.satmcarwls<-satMcarWLS(smoking.catdata)

smoking.E<-rbind(c(1,-1,0),c(0,1,-1))
smoking.A<-kronecker(kronecker(diag(2),smoking.E),smoking.E)

smoking.loglin2.marhybrid<-funlinWLS(model=c("lin","log"),
obj=smoking.satmarml, A1=smoking.A, XL=rep(1,8))

smoking.loglin2.mcarhybrid<-funlinWLS(model=c("lin","log"),
obj=smoking.satmcarml, A1=smoking.A, XL=rep(1,8))
```

```
smoking.loglin2.mcarwls<-funlinWLS(model=c("lin","log"),
obj=smoking.satmcarwls, A1=smoking.A, XL=rep(1,8))

#MNAR example
#Minus log-likelihood of the MNAR described in the last paragraph of Section 3
mnar.mll<-function(p,fr){
 # p=(pi11(1),...,pi33(2),a2(11),a2(21),a2(31),a3(11),
 #                        a3(21),a3(31),a2(12),a2(22),
 #                        a2(32),a3(12),a3(22),a3(32))

 pi11.1<-p[1]; pi12.1<-p[2]; pi13.1<-p[3]
 pi21.1<-p[4]; pi22.1<-p[5]; pi23.1<-p[6]
 pi31.1<-p[7]; pi32.1<-p[8]
 pi33.1=1-pi11.1-pi12.1-pi13.1-pi21.1-pi22.1-pi23.1-pi31.1-pi32.1

 pi11.2<-p[9]; pi12.2<-p[10];pi13.2<-p[11]
 pi21.2<-p[12];pi22.2<-p[13];pi23.2<-p[14]
 pi31.2<-p[15];pi32.2<-p[16]
 pi33.2=1-pi11.2-pi12.2-pi13.2-pi21.2-pi22.2-pi23.2-pi31.2-pi32.2

 a2.1.1<-p[17];a2.2.1<-p[18];a2.3.1<-p[19]
 a3.1.1<-p[20];a3.2.1<-p[21];a3.3.1<-p[22]

 a2.1.2<-p[23];a2.2.2<-p[24];a2.3.2<-p[25]
 a3.1.2<-p[26];a3.2.2<-p[27];a3.3.2<-p[28]

 value<- -(
  fr[1,1]*log(pi11.1*(1-a2.1.1-a3.1.1))+fr[1,2]*log(pi12.1*(1-a2.2.1-a3.1.1))+
  fr[1,3]*log(pi13.1*(1-a2.3.1-a3.1.1))+
  fr[1,4]*log(pi21.1*(1-a2.1.1-a3.2.1))+fr[1,5]*log(pi22.1*(1-a2.2.1-a3.2.1))+
  fr[1,6]*log(pi23.1*(1-a2.3.1-a3.2.1))+
  fr[1,7]*log(pi31.1*(1-a2.1.1-a3.3.1))+fr[1,8]*log(pi32.1*(1-a2.2.1-a3.3.1))+
  fr[1,9]*log(pi33.1*(1-a2.3.1-a3.3.1))+

  fr[1,10]*log(pi11.1*a2.1.1 + pi12.1*a2.2.1 + pi13.1*a2.3.1)+
  fr[1,11]*log(pi21.1*a2.1.1 + pi22.1*a2.2.1 + pi23.1*a2.3.1)+
  fr[1,12]*log(pi31.1*a2.1.1 + pi32.1*a2.2.1 + pi33.1*a2.3.1)+

  fr[1,13]*log(pi11.1*a3.1.1 + pi21.1*a3.2.1 + pi31.1*a3.3.1)+
  fr[1,14]*log(pi12.1*a3.1.1 + pi22.1*a3.2.1 + pi32.1*a3.3.1)+
  fr[1,15]*log(pi13.1*a3.1.1 + pi23.1*a3.2.1 + pi33.1*a3.3.1)+

  fr[2,1]*log(pi11.2*(1-a2.1.2-a3.1.2))+fr[2,2]*log(pi12.2*(1-a2.2.2-a3.1.2))+
  fr[2,3]*log(pi13.2*(1-a2.3.2-a3.1.2))+
  fr[2,4]*log(pi21.2*(1-a2.1.2-a3.2.2))+fr[2,5]*log(pi22.2*(1-a2.2.2-a3.2.2))+
  fr[2,6]*log(pi23.2*(1-a2.3.2-a3.2.2))+
  fr[2,7]*log(pi31.2*(1-a2.1.2-a3.3.2))+fr[2,8]*log(pi32.2*(1-a2.2.2-a3.3.2))+
  fr[2,9]*log(pi33.2*(1-a2.3.2-a3.3.2))+

  fr[2,10]*log(pi11.2*a2.1.2 + pi12.2*a2.2.2 + pi13.2*a2.3.2)+
  fr[2,11]*log(pi21.2*a2.1.2 + pi22.2*a2.2.2 + pi23.2*a2.3.2)+
  fr[2,12]*log(pi31.2*a2.1.2 + pi32.2*a2.2.2 + pi33.2*a2.3.2)+
```

```
  fr[2,13]*log(pi11.2*a3.1.2 + pi21.2*a3.2.2 + pi31.2*a3.3.2)+
  fr[2,14]*log(pi12.2*a3.1.2 + pi22.2*a3.2.2 + pi32.2*a3.3.2)+
  fr[2,15]*log(pi13.2*a3.1.2 + pi23.2*a3.2.2 + pi33.2*a3.3.2)

 )
value
}

mnar.fit<-constrOptim(theta=c(rep(1/9,16), rep(1/3,12)), f=mnar.mll,
method="Nelder-Mead", ui=diag(28), ci=rep(0,28),
control=list(maxit=10000), outer.iterations=1000, fr=smoking.TF)

#hessian matrix
mnar.der<-deriv3(~-(
 o1.1*log(pi11.1*(1-a2.1.1-a3.1.1))+o1.2*log(pi12.1*(1-a2.2.1-a3.1.1))+
 o1.3*log(pi13.1*(1-a2.3.1-a3.1.1))+
 o1.4*log(pi21.1*(1-a2.1.1-a3.2.1))+o1.5*log(pi22.1*(1-a2.2.1-a3.2.1))+
 o1.6*log(pi23.1*(1-a2.3.1-a3.2.1))+
 o1.7*log(pi31.1*(1-a2.1.1-a3.3.1))+o1.8*log(pi32.1*(1-a2.2.1-a3.3.1))+
    o1.9*log((1-pi11.1-pi12.1-pi13.1-pi21.1-
 pi22.1-pi23.1-pi31.1-pi32.1)*(1-a2.3.1-a3.3.1))+
 o1.10*log(pi11.1*a2.1.1 + pi12.1*a2.2.1 + pi13.1*a2.3.1)+
 o1.11*log(pi21.1*a2.1.1 + pi22.1*a2.2.1 + pi23.1*a2.3.1)+
 o1.12*log(pi31.1*a2.1.1 + pi32.1*a2.2.1 +
      (1-pi11.1-pi12.1-pi13.1-pi21.1-pi22.1-pi23.1-pi31.1-pi32.1)*a2.3.1)+
 o1.13*log(pi11.1*a3.1.1 + pi21.1*a3.2.1 + pi31.1*a3.3.1)+
 o1.14*log(pi12.1*a3.1.1 + pi22.1*a3.2.1 + pi32.1*a3.3.1)+
 o1.15*log(pi13.1*a3.1.1 + pi23.1*a3.2.1 +
(1-pi11.1-pi12.1-pi13.1-pi21.1-pi22.1-pi23.1-pi31.1-pi32.1)*a3.3.1)+
 o2.1*log(pi11.2*(1-a2.1.2-a3.1.2))+o2.2*log(pi12.2*(1-a2.2.2-a3.1.2))+
 o2.3*log(pi13.2*(1-a2.3.2-a3.1.2))+
 o2.4*log(pi21.2*(1-a2.1.2-a3.2.2))+o2.5*log(pi22.2*(1-a2.2.2-a3.2.2))+
 o2.6*log(pi23.2*(1-a2.3.2-a3.2.2))+
 o2.7*log(pi31.2*(1-a2.1.2-a3.3.2))+o2.8*log(pi32.2*(1-a2.2.2-a3.3.2)) +
o2.9*log((1-pi11.2-pi12.2-pi13.2-pi21.2-
  pi22.2-pi23.2-pi31.2-pi32.2)*(1-a2.3.2-a3.3.2))+
 o2.10*log(pi11.2*a2.1.2 + pi12.2*a2.2.2 + pi13.2*a2.3.2)+
 o2.11*log(pi21.2*a2.1.2 + pi22.2*a2.2.2 + pi23.2*a2.3.2)+
 o2.12*log(pi31.2*a2.1.2 + pi32.2*a2.2.2 +
(1-pi11.2-pi12.2-pi13.2-pi21.2-pi22.2-pi23.2-pi31.2-pi32.2)*a2.3.2)+
 o2.13*log(pi11.2*a3.1.2 + pi21.2*a3.2.2 + pi31.2*a3.3.2)+
 o2.14*log(pi12.2*a3.1.2 + pi22.2*a3.2.2 + pi32.2*a3.3.2)+
 o2.15*log(pi13.2*a3.1.2 + pi23.2*a3.2.2 +
(1-pi11.2-pi12.2-pi13.2-pi21.2-pi22.2-pi23.2-pi31.2-pi32.2)*a3.3.2)

),c("pi11.1","pi12.1","pi13.1","pi21.1","pi22.1","pi23.1","pi31.1","pi32.1",
 "pi11.2","pi12.2","pi13.2","pi21.2","pi22.2","pi23.2","pi31.2","pi32.2",
 "a2.1.1","a2.2.1","a2.3.1","a3.1.1","a3.2.1","a3.3.1","a2.1.2","a2.2.2",
 "a2.3.2","a3.1.2","a3.2.2","a3.3.2"),
c("pi11.1","pi12.1","pi13.1","pi21.1","pi22.1","pi23.1","pi31.1","pi32.1",
 "pi11.2","pi12.2","pi13.2","pi21.2","pi22.2","pi23.2","pi31.2","pi32.2",
 "a2.1.1","a2.2.1","a2.3.1","a3.1.1","a3.2.1","a3.3.1","a2.1.2","a2.2.2",
```

```
 "a2.3.2","a3.1.2","a3.2.2","a3.3.2",
 "o1.1","o1.2","o1.3","o1.4","o1.5","o1.6","o1.7","o1.8","o1.9","o1.10",
 "o1.11","o1.12","o1.13","o1.14","o1.15",
 "o2.1","o2.2","o2.3","o2.4","o2.5","o2.6","o2.7","o2.8","o2.9","o2.10",
 "o2.11","o2.12","o2.13","o2.14","o2.15")
)

p<-mnar.fit$par;TF<-smoking.TF
mnar.InfObs<-mnar.der(p[1],p[2],p[3],p[4],p[5],p[6],p[7],p[8],p[9],p[10],
p[11],p[12],p[13],p[14], p[15],p[16],p[17],p[18],p[19],p[20],p[21],
p[22],p[23],p[24],p[25],p[26],p[27],p[28], TF[1,1],TF[1,2],TF[1,3],
TF[1,4],TF[1,5],TF[1,6],TF[1,7],TF[1,8],TF[1,9],TF[1,10],
TF[1,11],TF[1,12],TF[1,13],TF[1,14],TF[1,15],
TF[2,1],TF[2,2],TF[2,3],TF[2,4],TF[2,5],TF[2,6],TF[2,7],TF[2,8],TF[2,9],TF[2,10],
TF[2,11],TF[2,12],TF[2,13],TF[2,14],TF[2,15])
b<-smoking.catdata$b #b in (8), i.e., rep(1,2)%x%c(rep(0,8),1)
B<-smoking.catdata$B #B in (8), i.e., diag(2)%x%rbind(diag(8),rep(-1,8))

smoking.loglin2mnar.hybrid<-funlinWLS(model=c("lin","log"),
theta=as.vector(b+c(B%*%mnar.fit$par[1:16])),
Vtheta=B%*%solve(attr(mnar.InfObs,"hessian")[1,,])[1:16,1:16]%*%t(B),
A1=smoking.A,X=rep(1,8))
```

`linML` *Fitting Linear Models via Maximum Likelihood*

Description

`linML` fits linear models by ML (maximum likelihood). For complete data, it is based on a object of the class `readCatdata`. For missing data, it is based on a object of the class `satMarML` (under MAR or MCAR). Depending on the formulation (freedom equations or constraints), different arguments must be informed.

Usage

```
linML(obj, A, X, U, start, maxit=100, trace=0, epsilon1=1e-6,
epsilon2=1e-6, zeroN, digits)
```

Arguments

`obj`	object of the class `readCatdata` (for complete data) or `satMarML` (for missing data).
`A`	a matrix that specifies the linear functions of the probabilities to be modeled; by default, it is diag(*S*) %x% cbind(diag(*R-1*),rep(*0*,*R-1*)), which discards the last element of the probability vector associated to each multinomial, where *S* represents the number of subpopulations and *R*, the number of response categories.
`X`	a model specification matrix for the freedom equation formulation.
`U`	a matrix for the constraint formulation.

`start`	by default, the function uses the proportions of the complete data as starting values in the iterative process, but the current argument allows the user to inform an alternative starting value for the parameters of the model if the freedom equation formulation is considered and the matrix *A* is modeling `S*(R-1)` linear functions; a vector with these values must be informed.
`maxit`	the maximum number of iterations (the default is 100).
`trace`	the alternatives are: *0* for no printing (default), *1* for showing only the value of the likelihood ratio statistics at each iteration of the iterative process, and *2* for including the parameter estimates at each iteraction.
`epsilon1`	the convergence criterion of the iterative process is attained if the absolute difference of the values of the likelihood ratio statistic of successive iterations is less than the value defined in `epsilon1`, `1e-6` by default.
`epsilon2`	the convergence criterion of the iterative process is attained if the absolute differences of the values of estimates for all parameters of the marginal probabilities of categorization in consecutive iterations are less than the value defined in `epsilon2`, `1e-6` by default.
`zeroN`	values used to replace null frequencies in the denominator of the Neyman statistic; by default, the function replaces the values by `1/(R*nst)`, where *nst* is the sample size of the missingness pattern associated to the corresponding subpopulation; the user may indicate alternative values in a matrix with *S* rows and an additional column relatively to the number of columns of *Rp*; the first column relates to the completely categorized "missingness" patterns, and the remaining columns to the other missingness patterns as they appear in *Rp*; the values must be non-negative and less or equal to 0.5.
`digits`	integer value indicating the number of decimal places to round results when shown by `print` and `summary`; this argument works also when specified directly in both generic functions; default value is 4.

Details

Linear models may be fit to the functions A%*%Theta using a freedom equation formulation A%*%Theta=X%*%Beta, where Beta are the parameters to be estimated, or using a constraint formulation U%*%A%*%Theta=0. Both formulations lead to an equivalent model fit if U%*%X=0.

The generic functions `print` and `summary` are used to print the results and to obtain a summary thereof.

Value

An object of the class `linML` is a list containing most of the components of the argument `obj` as well as the following components:

`thetaH`	vector of ML estimates for all product-multinomial probabilities under the linear model for the marginal probabilities of categorization and, in the case of missing data, under an assumption of an ignorable missingness mechanism.
`VthetaH`	corresponding estimated covariance matrix.
`beta`	vector of ML estimates for the parameters of the linear model (only for freedom equation formulation).

`Vbeta`	corresponding estimated covariance matrix (only for freedom equation formulation).
`Fu`	observed linear functions, without model constraints.
`VFu`	corresponding estimated covariance matrix.
`FH`	ML estimates for the linear functions under the fitted model.
`VFH`	corresponding estimated covariance matrix.
`QvH`	likelihood ratio statistic for testing the goodness of fit of the linear model (for missing data, conditional on the assumed missingness mechanism).
`QpH`	Pearson statistic for testing the goodness of fit of the linear model (for missing data, conditional on the assumed missingness mechanism).
`QnH`	Neyman statistic for testing the goodness of fit of the linear model (for missing data, conditional on the assumed missingness mechanism).
`QwH`	Wald statistic for testing the goodness of fit of the linear model (for missing data, conditional on the assumed missingness mechanism).
`glH`	degrees of freedom for testing the goodness of fit of the linear model (for missing data, conditional on the assumed missingness mechanism).
`QvHMCAR`	likelihood ratio statistic for the conditional test of both the linear model and MCAR given a MAR assumption (for missing data only).
`QpHMCAR`	Pearson statistic for the conditional test of both the linear model and MCAR given a MAR assumption (for missing data only).
`QnHMCAR`	Neyman statistic for the conditional test of both the linear model and MCAR given a MAR assumption (for missing data only).
`glHMCAR`	degrees of freedom for the conditional test of both the linear model and MCAR given a MAR assumption (for missing data only).
`ystH`	for complete data, it contains the ML estimates for the frequencies under the linear model; for missing data, it contains the ML estimates for the augmented frequencies under both the linear model and the assumed missingness mechanism.

Author(s)

Frederico Zanqueta Poleto(`<frederico@poleto.com>`)
Julio da Motta Singer (`<jmsinger@ime.usp.br>`)
Carlos Daniel Paulino (`<daniel.paulino@math.ist.utl.pt>`)
with the collaboration of
Fabio Mathias Correa (`<fmcorrea@uesc.br>`)
Enio Galinkin Jelihovschi (`<eniojelihovs@gmail.com>`)

References

Paulino, C.D. e Singer, J.M. (2006). *Analise de dados categorizados* (in Portuguese). Sao Paulo: Edgard Blucher.

Poleto, F.Z. (2006). *Analise de dados categorizados com omissao* (in Portuguese). Dissertacao de mestrado. IME-USP. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2007). *Analyzing categorical data with complete or missing responses using the Catdata package*. Unpublished vignette. http://www.poleto.com/missing.html.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2012). A product-multinomial framework for categorical data analysis with missing responses. To appear in *Brazilian Journal of Probability and Statistics*. http://imstat.org/bjps/papers/BJPS198.pdf.

Singer, J. M., Poleto, F. Z. and Paulino, C. D. (2007). Catdata: software for analysis of categorical data with complete or missing responses. *Actas de la XII Reunion Cientifica del Grupo Argentino de Biometria y I Encuentro Argentino-Chileno de Biometria*. http://www.poleto.com/SingerPoletoPaulino2007GAB.pdf.

Examples

```
#Example 8.1 of Paulino and Singer (2006)

e81.TF<-c(192,1,5,2,146,5,11,12,71)
e81.catdata<-readCatdata(TF=e81.TF)

e81.U<-rbind(c(0,-1, 0,1,0, 0,0,0),
 c(0, 0,-1,0,0, 0,1,0),
 c(0, 0, 0,0,0,-1,0,1))

e81.X<-rbind(c(1,0,0,0,0),
 c(0,1,0,0,0),
 c(0,0,1,0,0),
 c(0,1,0,0,0),
 c(0,0,0,1,0),
 c(0,0,0,0,1),
 c(0,0,1,0,0),
 c(0,0,0,0,1))

#Two equivalent ways of fitting the same symmetry model

e81.linml1<-linML(e81.catdata,U=e81.U)
e81.linml2<-linML(e81.catdata,X=e81.X)
e81.linml1 #constraint formulation
e81.linml2 #freedom equation formulation
summary(e81.linml1)

#Example 8.2 of Paulino and Singer (2006)
e82.TF<-c(11,5,0,14,34,7,2,13,11)

e82.catdata<-readCatdata(TF=e82.TF)

e82.U<-rbind(c(0, 1,1,-1,0,0,-1, 0),
 c(0,-1,0, 1,0,1, 0,-1))
e82.X<-rbind(c(1, 0, 0,0,0,0),
 c(0, 1, 0,0,0,0),
 c(0,-1, 1,0,1,0),
 c(0, 0, 1,0,0,0),
 c(0, 0, 0,1,0,0),
```

```
 c(0, 1,-1,0,0,1),
 c(0, 0, 0,0,1,0),
 c(0, 0, 0,0,0,1))

e82.linml1<-linML(e82.catdata,U=e82.U)

e82.linml2<-linML(e82.catdata,X=e82.X)

e82.A<-rbind(c(1,1,1,0,0,0,0,0,0),
 c(0,0,0,1,1,1,0,0,0),
 c(1,0,0,1,0,0,1,0,0),
 c(0,1,0,0,1,0,0,1,0))

e82.U2<-rbind(c(1,0,-1, 0),
  c(0,1, 0,-1))

e82.X2<-rbind(c(1,0),
  c(0,1),
  c(1,0),
  c(0,1))

e82.linml3<-linML(e82.catdata,A=e82.A,U=e82.U2)
e82.linml4<-linML(e82.catdata,A=e82.A,X=e82.X2)

#Four equivalent ways of fitting the same marginal homogeneity model
e82.linml1;e82.linml2;e82.linml3;e82.linml4

#Example 13.2 of Paulino and Singer (2006)

e132.TF2<-c(7,11,2,3,9,5,1e-5,10,4, 8,7,3,0, 0,7,14,7) #replace zero by small value
e132.Zp<-cbind(rbind(cbind(
   kronecker(rep(1,2),diag(3)),rep(0,6)),
     cbind(matrix(0,3,3),rep(1,3)) ),
   rbind(cbind(rep(1,3),matrix(0,3,3)),
   cbind(rep(0,6),kronecker(rep(1,2),diag(3)))))
e132.Rp<-c(4,4)
e132.catdata2<-readCatdata(TF=e132.TF2,Zp=e132.Zp,Rp=e132.Rp)

e132.satmarml2<-satMarML(e132.catdata2)

e132.U<-rbind(c(0, 1,1,-1,0,0,-1, 0),
  c(0,-1,0, 1,0,1, 0,-1) )

e132.linml<-linML(e132.satmarml2,U=e132.U)

#Example 2 of Poleto et al (2012)
obes.TF<-rbind(
   c( 90, 9, 3, 7, 0,1, 1, 8,16, 5,0, 0, 9,3,0,0,129,18,6,13,32, 5,33,11,70,24),
   c(150,15, 8, 8, 8,9, 7,20,38, 3,1,11,16,6,1,3, 42, 2,3,13,45, 7,33, 4,55,14),
   c(152,11, 8,10, 7,7, 9,25,48, 6,2,14,13,5,0,3, 36, 5,4, 3,59,17,31, 9,40, 9),
   c(119, 7, 8, 3,13,4,11,16,42, 4,4,13,14,2,1,4, 18, 3,3, 1,82,24,23, 6,37,14),
   c(101, 4, 2, 7, 8,0, 6,15,82, 9,8,12, 6,1,0,1, 13, 1,2, 2,95,23,34,12,15, 3),
   c( 75, 8, 2, 4, 2,2, 1, 8,20, 0,0, 4, 7,2,0,1,109,22,7,24,23, 5,27, 5,65,19),
```

```
   c(154,14,13,19, 2,6, 6,21,25, 3,1,11,16,3,0,4, 47, 4,1, 8,47, 7,23, 5,39,13),
   c(148, 6,10, 8,12,0, 8,27,36, 0,7,17, 8,1,1,4, 39, 6,7,13,53,16,25, 9,23, 8),
   c(129, 8, 7, 9, 6,2, 7,14,36, 9,4,13,31,4,2,6, 19, 1,2, 2,58,37,21, 1,23,10),
   c( 91, 9, 5, 3, 6,0, 6,15,83,15,6,23, 5,0,0,1, 11, 1,2, 3,89,32,43,15,14, 5))


obes.Zp<-kronecker(t(rep(1,10)),
   cbind(kronecker(diag(4),rep(1,2)),
 kronecker(diag(2),kronecker(rep(1,2),diag(2))),
 kronecker(rep(1,2),diag(4)),
 kronecker(diag(2),rep(1,4)),
 kronecker(rep(1,2),kronecker(diag(2),rep(1,2))),
 kronecker(rep(1,4),diag(2))))

obes.Rp<-kronecker(rep(1,10),t(c(4,4,4,2,2,2)))
obes.catdata<-readCatdata(TF=obes.TF,Zp=obes.Zp,Rp=obes.Rp)
obes.mar<-satMarML(obes.catdata)

obes.A.marg <- kronecker(diag(10),t(cbind(
       kronecker(diag(2),rep(1,4)),
   kronecker(rep(1,2),kronecker(diag(2),rep(1,2))),
   kronecker(rep(1,4),diag(2))))[c(2,4,6),])

obes.age<-c(6,8,10,8,10,12,10,12,14,12,14,16,14,16,18)
obes.X2<-kronecker(diag(2),cbind(rep(1,15),obes.age,obes.age^2))

# Not run
# obes.lin2.ml<-linML(obes.mar,A=obes.A.marg,X=obes.X2)

obesR.TF<-obes.TF

obesR.TF[obesR.TF==0]<-1e-6 #Replacing null frequencies by 10^{-6}

obesR.catdata<-readCatdata(TF=obesR.TF,Zp=obes.Zp,Rp=obes.Rp)
obesR.mar<-satMarML(obesR.catdata)
obesR.lin2.ml<-linML(obesR.mar,A=obes.A.marg,X=obes.X2)

obesR.lin2.ml
```

`loglinML` *Fitting Log-Linear Models via Maximum Likelihood*

Description

`loglinML` fits log-linear models by ML (maximum likelihood). For complete data, it is based on a object of the class `readCatdata`. For missing data, it is based on a object of the class `satMarML` (under MAR or MCAR). Depending on the formulation (freedom equations or constraints) and on the model type (ordinary or generalized log-linear model), different arguments must be informed.

Usage

```
loglinML(obj, A, X, U, XL, UL, start, maxit=100, trace=0, epsilon1=1e-6,
epsilon2=1e-6, zeroN, digits)
```

Arguments

`obj`	object of the class `readCatdata` (for complete data) or `satMarML` (for missing data).
`A`	a matrix that specifies the log-linear functions of the probabilities to be modeled; by default, it is diag(*S*) %x% cbind(diag(*R-1*),rep(*-1*,*R-1*)), which generates logits with the last (R) category as baseline.
`X`	a model specification matrix for the freedom equation formulation of the ordinary log-linear model.
`U`	a matrix for the constraint formulation of the ordinary log-linear specification.
`XL`	a model specification matrix for the freedom equation formulation of the generalized log-linear model.
`UL`	a matrix for the constraint formulation of the generalized log-linear model.
`start`	by default, the function uses the proportions of the complete data as starting values in the iterative process, but the current argument allows the user to inform an alternative starting value for the parameters of the model if the freedom equation formulation is considered and the matrix *A* is modeling `S*(R-1)` log-linear functions; a vector with these values must be informed.
`maxit`	the maximum number of iterations (the default is 100).
`trace`	the alternatives are: *0* for no printing (default), *1* for showing only the value of the likelihood ratio statistics at each iteration of the iterative process, and *2* for including also the parameter estimates at each iteraction.
`epsilon1`	the convergence criterion of the iterative process is attained if the absolute difference of the values of the likelihood ratio statistic in successive iterations is less than the value defined in `epsilon1`, `1e-6` by default.
`epsilon2`	the convergence criterion of the iterative process is attained if the absolute differences of the values of estimates for all parameters of the marginal probabilities of categorization in consecutive iterations are less than the value defined in `epsilon2`, `1e-6` by default.
`zeroN`	values used to replace null frequencies in the denominator of the Neyman statistic; by default, the function replaces the values by `1/(R*nst)`, where *nst* is the sample size of the missingness pattern associated to the corresponding subpopulation; the user may indicate alternative values in a matrix with *S* rows and an additional column relatively to the number of columns of *Rp*; the first column relates to the completely categorized "missingness" patterns, and the remaining columns to the other missingness patterns as they appear in *Rp*; the values must be non-negative and less or equal to 0.5.
`digits`	integer value indicating the number of decimal places to round results when shown by `print` and `summary`; this argument works also when specified directly in both generic functions; default value is 4.

Details

`loglinML` handles both ordinary and generalized log-linear model types either under a freedom equation formulation or under a constraint formulation. X and U are used for ordinary log-linear models, and XL and UL are used for generalized log-linear models. X and XL are used for the freedom equation formulation, and U and UL are used for the constraint formulation. Namely, the 4 ways with which the function allows to specify the model are: log(Theta)=nu+X%*%Beta, U%*%log(Theta)=0, A%*%log(Theta)=XL%*%Beta, UL%*%A%*%log(Theta)=0, where nu are non-estimated parameters included only to satisfy the natural constraints of the product-multinomial distribution and Beta are the parameters to be estimated.

The generic functions `print` and `summary` are used to print the results and to obtain a summary thereof.

Value

An object of the class `loglinML` is a list containing most of the components of the argument `obj` as well as the following components:

`thetaH`	vector of ML estimates for all product-multinomial probabilities under the log-linear model for the marginal probabilities of categorization and, in the case of missing data, under an assumption of an ignorable missingness mechanism.
`VthetaH`	corresponding estimated covariance matrix.
`beta`	vector of ML estimates for the parameters of the log-linear model (only for freedom equation formulation).
`Vbeta`	corresponding estimated covariance matrix (only for the freedom equation formulation).
`Fu`	observed log-linear functions, without model constraints.
`VFu`	corresponding estimated covariance matrix.
`FH`	ML estimates for the log-linear functions under the fitted model.
`VFH`	corresponding estimated covariance matrix.
`QvH`	likelihood ratio statistic for testing the goodness of fit of the log-linear model (for missing data, conditional on the assumed missingness mechanism).
`QpH`	Pearson statistic for testing the goodness of fit of the log-linear model (for missing data, conditional on the assumed missingness mechanism).
`QnH`	Neyman statistic for testing the goodness of fit of the log-linear model (for missing data, conditional on the assumed missingness mechanism).
`QwH`	Wald statistic for testing the goodness of fit of the log-linear model (for missing data, conditional on the assumed missingness mechanism).
`glH`	degrees of freedom for testing the goodness of fit of the log-linear model (for missing data, conditional on the assumed missingness mechanism).
`QvHMCAR`	likelihood ratio statistic for the conditional test of both the log-linear model and MCAR given a MAR assumption (for missing data only).
`QpHMCAR`	Pearson statistic for the conditional test of both the log-linear model and MCAR given a MAR assumption (for missing data only).

`QnHMCAR`	Neyman statistic for the conditional test of both the log-linear model and MCAR given a MAR assumption (for missing data only).
`glHMCAR`	degrees of freedom for the conditional test of both the log-linear model and MCAR given a MAR assumption (for missing data only).
`ystH`	for complete data, it has the ML estimates for the frequencies under the log-linear model; for missing data, it has the ML estimates for the augmented frequencies under both the log-linear model and the assumed missingness mechanism.

Author(s)

Frederico Zanqueta Poleto(`<frederico@poleto.com>`)
Julio da Motta Singer (`<jmsinger@ime.usp.br>`)
Carlos Daniel Paulino (`<daniel.paulino@math.ist.utl.pt>`)
with the collaboration of
Fabio Mathias Correa (`<fmcorrea@uesc.br>`)
Enio Galinkin Jelihovschi (`<eniojelihovs@gmail.com>`)

References

Paulino, C.D. e Singer, J.M. (2006). *Analise de dados categorizados* (in Portuguese). Sao Paulo: Edgard Blucher.

Poleto, F.Z. (2006). *Analise de dados categorizados com omissao* (in Portuguese). Dissertacao de mestrado. IME-USP. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2007). *Analyzing categorical data with complete or missing responses using the Catdata package*. Unpublished vignette. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2012). A product-multinomial framework for categorical data analysis with missing responses. To appear in *Brazilian Journal of Probability and Statistics*. `http://imstat.org/bjps/papers/BJPS198.pdf`.

Singer, J. M., Poleto, F. Z. and Paulino, C. D. (2007). Catdata: software for analysis of categorical data with complete or missing responses. *Actas de la XII Reunion Cientifica del Grupo Argentino de Biometria y I Encuentro Argentino-Chileno de Biometria*. `http://www.poleto.com/SingerPoletoPaulino2007GAB.pdf`.

Examples

```
#Example 9.1 of Paulino and Singer (2006)
e91.TF<-c(3,25,32,68)
e91.catdata<-readCatdata(TF=e91.TF)
e91.U<-c(1,-1,-1,1)

e91.X<-rbind(c(0,0), c(0,1),
             c(1,0), c(1,1))

e91.X2<-rbind(c(0,0,0),
               c(0,1,0),
```

```
               c(1,0,0),
              c(1,1,1))

e91.loglinml1<-loglinML(e91.catdata,U=e91.U)
e91.loglinml2<-loglinML(e91.catdata,X=e91.X)
e91.loglinml3<-loglinML(e91.catdata,X=e91.X2)
e91.loglinml4<-loglinML(e91.catdata,A=c(1,-1,-1,1),XL=1)

# Independence ordinary log-linear model, constraint formulation
e91.loglinml1
# Independence ordinary log-linear model, freedom equation formulation
e91.loglinml2
#Saturated ordinary log-linear model, freedom equation formulation
e91.loglinml3

#Saturated generalized log-linear model, freedom equation formulation
e91.loglinml4

#95% confidence interval for log-odds ratio and for odds ratio
round(e91.loglinml4$beta+c(-1,1)*qnorm(0.975)*sqrt(e91.loglinml4$Vbeta),3)
round(exp(e91.loglinml4$beta),3)
round(exp(e91.loglinml4$beta+c(-1,1)*qnorm(0.975)*sqrt(e91.loglinml4$Vbeta)),3)

#Example 1 of Poleto et al (2012)
smoking.TF<-rbind(c(167,17,19,10,1,3,52,10,11, 176,24,121, 28,10,12),
                  c(120,22,19, 8,5,1,39,12,12, 103, 3, 80, 31, 8,14))

smoking.Zp<-t(rep(1,2))%x%cbind(diag(3)%x%rep(1,3), rep(1,3)%x%diag(3))

smoking.Rp<-rbind(c(3,3),c(3,3))

smoking.catdata<-readCatdata(TF=smoking.TF,Zp=smoking.Zp,Rp=smoking.Rp)
smoking.catdata #Proportions of the complete data

smoking.satmarml<-satMarML(smoking.catdata)
smoking.satmcarml<-satMarML(smoking.catdata,missing="MCAR")

smoking.E<-rbind(c(1,-1,0),c(0,1,-1))

smoking.A<-diag(2)%x%smoking.E%x%smoking.E
smoking.loglin2.marml<-loglinML(smoking.satmarml,A=smoking.A,XL=rep(1,8))
smoking.loglin2.mcarml<-loglinML(smoking.satmcarml,A=smoking.A,XL=rep(1,8))
```

`readCatdata` *Reads Categorical Data*

Description

`readCatdata` inputs the categorical data, accommodating complete or missing observations. This corresponds to objects of the classes `vector` or `matrix` which represent a table of frequencies assumed to follow a product-multinomial distribution. For complete data, only the argument `TF` is required. Linear, log-linear and functional linear models may be subsequently fitted, respectively, using the functions `linML()`, `loglinML()` and `funlinWLS()`. For missing data, all arguments `TF`, `Zp` and `Rp` are required. Before proceeding to model fitting, inferences for the saturated product-multinomial parameters are conducted using either the function `satMarML()` or `satMcarWLS()`.

Usage

```
readCatdata(TF, Zp, Rp)
```

Arguments

`TF`	a matrix including the table of frequencies (see details).
`Zp`	a matrix with indicators of the partially classified data (see details).
`Rp`	a matrix with the number of response classes corresponding to the missingness patterns (see details).

Details

Whenever `TF` is a vector it represents one population assumed to follow a multinomial distribution. Whenever `TF` is a matrix, each row represents a subpopulation, the set of which is assumed to follow a product-multinomial distribution. For complete data, `TF` is usually a simple matrix (table of frequencies) with S rows (subpopulations) and R columns (response categories). For missing categorical data, each row of `TF` is assumed to have first the R frequencies associated to the response categories of the fully classified data, and thereafter the frequencies associated to the response classes of the partially classified data. The term response class is used to indicate that the corresponding observed data may only be classified into a set of response categories and not to the individual categories within this set. This is informed via indicator vectors with R rows, the elements of which are equal to 1 for the positions corresponding to the response categories in the response class and to 0 otherwise. For computational simplicity, it is assumed that the response classes corresponding to each missingness pattern belong to a partition of the response categories. `Zp` contains all the indicator vectors, which were first combined columnwise within subpopulations (without mixing different missingness patterns) and then between subpopulations. As `Rp` is a matrix that contains the number of response classes of each missingness pattern (column) of each subpopulation (row), `Zp` is a matrix with R rows and `sum(Rp)` columns. If the subpopulations do not have the same number of missingness patterns and/or response classes, the matrices `TF` and `Rp` shall be completed with "0". The generic functions `print` and `summary` are used to print the results and to obtain a summary thereof. The latter is particularly useful for checking whether the missing categorical data input was conducted in the right way.

Value

An object of the class `readCatdata` is a list containing at least the following components:

`R`	number of response categories.
`S`	number of subpopulations.
`Tt`	number of missingness patterns of each subpopulation.
`Nst`	frequencies under each missingness pattern of each subpopulation.

`nstm`	sample size of each missingness pattern of each subpopulation.
`nsmm`	sample size of each subpopulation.
`pst`	vectors of proportions computed under each missingness pattern of each subpopulation.
`Vpst`	covariance matrices of the proportions.
`theta`	vector of estimates for all product-multinomial probabilities under the saturated model (for complete data only).
`Vtheta`	corresponding covariance matrix (only for complete data).

Author(s)

Frederico Zanqueta Poleto(<`frederico@poleto.com`>)
Julio da Motta Singer (<`jmsinger@ime.usp.br`>)
Carlos Daniel Paulino (<`daniel.paulino@math.ist.utl.pt`>)
with the collaboration of
Fabio Mathias Correa (<`fmcorrea@uesc.br`>)
Enio Galinkin Jelihovschi (<`eniojelihovs@gmail.com`>)

References

Paulino, C.D. e Singer, J.M. (2006). *Analise de dados categorizados* (in Portuguese). Sao Paulo: Edgard Blucher.

Poleto, F.Z. (2006). *Analise de dados categorizados com omissao* (in Portuguese). Dissertacao de mestrado. IME-USP. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2007). *Analyzing categorical data with complete or missing responses using the Catdata package*. Unpublished vignette. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2012). A product-multinomial framework for categorical data analysis with missing responses. To appear in *Brazilian Journal of Probability and Statistics*. `http://imstat.org/bjps/papers/BJPS198.pdf`.

Singer, J. M., Poleto, F. Z. and Paulino, C. D. (2007). Catdata: software for analysis of categorical data with complete or missing responses. *Actas de la XII Reunion Cientifica del Grupo Argentino de Biometria y I Encuentro Argentino-Chileno de Biometria*. `http://www.poleto.com/SingerPoletoPaulino2007GAB.pdf`.

Examples

```
#Example 1.5 of Paulino and Singer (2006)
#S=4 subpopulations, R=4 response categories, with complete data
e15.TF<-rbind(c(19, 5, 4, 2),
  c( 5, 8, 0,17),
  c(11, 6, 7, 6),
  c( 2, 5, 1,22))
e15.catdata<-readCatdata(TF=e15.TF)

e15.catdata #shows proportions and standard errors
```

```
#Example 13.4 of Paulino and Singer (2006)
#S=1 subpopulation, R=4 response categories, with missing data
#2 missingness patterns with 2 response classes each
e134.TF<-c(12,4,5,2, 50,31, 27,12)
e134.Zp<-cbind(kronecker(diag(2),rep(1,2)),kronecker(rep(1,2),diag(2)))
e134.Rp<-c(2,2)
e134.catdata<-readCatdata(TF=e134.TF,Zp=e134.Zp,Rp=e134.Rp)
e134.catdata #Proportions of the complete data
summary(e134.catdata) #A more detailed analysis of the missing data input

#Example 13.2 of Paulino and Singer (2006)
#S=1 subpopulation, R=9 response categories, with missing data
#2 missingness patterns with 4 response classes each
e132.TF<-c(7,11,2,3,9,5,0,10,4, 8,7,3,0, 0,7,14,7)
e132.Zp<-cbind(rbind(cbind(kronecker(rep(1,2),diag(3)),rep(0,6)),
 cbind(matrix(0,3,3),rep(1,3)) ),
     rbind(cbind(rep(1,3),matrix(0,3,3)),
 cbind(rep(0,6),kronecker(rep(1,2),diag(3)))))

e132.Rp<-c(4,4)

e132.catdata<-readCatdata(TF=e132.TF,Zp=e132.Zp,Rp=e132.Rp)
summary(e132.catdata)

#Example 1 of Poleto et al (2012)
#S=2 subpopulation, R=9 response categories, with missing data
#in each subpopulation: 2 missingness patterns with 3 response classes each
smoking.TF<-rbind(c(167,17,19,10,1,3,52,10,11, 176,24,121, 28,10,12),
  c(120,22,19, 8,5,1,39,12,12, 103, 3, 80, 31, 8,14))
smoking.Zp<-t(rep(1,2))%x%cbind(diag(3)%x%rep(1,3), rep(1,3)%x%diag(3))
smoking.Rp<-rbind(c(3,3),c(3,3))
smoking.catdata<-readCatdata(TF=smoking.TF,Zp=smoking.Zp,Rp=smoking.Rp)
smoking.catdata

#Example 2 of Poleto et al (2012)
#S=10 subpopulation, R=8 response categories, with missing data
#in each subpopulation: 6 missingness patterns, 3 patterns with 4 response
#classes each, and other 3 patterns with 2 response classes
obes.TF<-rbind(
c(90, 9, 3, 7, 0,1, 1, 8,16, 5,0, 0, 9,3,0,0,129,18,6,13,32, 5,33,11,70,24),
c(150,15, 8, 8, 8,9, 7,20,38, 3,1,11,16,6,1,3, 42, 2,3,13,45, 7,33, 4,55,14),
c(152,11, 8,10, 7,7, 9,25,48, 6,2,14,13,5,0,3, 36, 5,4, 3,59,17,31, 9,40, 9),
c(119, 7, 8, 3,13,4,11,16,42, 4,4,13,14,2,1,4, 18, 3,3, 1,82,24,23, 6,37,14),
c(101, 4, 2, 7, 8,0, 6,15,82, 9,8,12, 6,1,0,1, 13, 1,2, 2,95,23,34,12,15, 3),
c( 75, 8, 2, 4, 2,2, 1, 8,20, 0,0, 4, 7,2,0,1,109,22,7,24,23, 5,27, 5,65,19),
c(154,14,13,19, 2,6, 6,21,25, 3,1,11,16,3,0,4, 47, 4,1, 8,47, 7,23, 5,39,13),
c(148, 6,10, 8,12,0, 8,27,36, 0,7,17, 8,1,1,4, 39, 6,7,13,53,16,25, 9,23, 8),
c(129, 8, 7, 9, 6,2, 7,14,36, 9,4,13,31,4,2,6, 19, 1,2, 2,58,37,21, 1,23,10),
c(91, 9, 5, 3, 6,0, 6,15,83,15,6,23, 5,0,0,1, 11, 1,2, 3,89,32,43,15,14, 5))

obes.Zp<-t(rep(1,10))%x%cbind(diag(4)%x%rep(1,2),
  diag(2)%x%rep(1,2)%x%diag(2), rep(1,2)%x%diag(4),
```

```
  diag(2)%x%rep(1,4),rep(1,2)%x%diag(2)%x%rep(1,2), rep(1,4)%x%diag(2))

obes.Rp<-rep(1,10)%x%t(c(4,4,4,2,2,2))
obes.catdata<-readCatdata(TF=obes.TF,Zp=obes.Zp,Rp=obes.Rp)
obes.catdata #Proportions of the complete data
```

`satMarML` *Fitting Saturated Models for the Marginal Probabilities of Categorization via Maximum Likelihood under MAR and MCAR assumption*

Description

`satMarML` fits saturated models for the marginal probabilities of categorization as well as missing at random (MAR) or missing completely at random (MCAR) models for the missingness mechanism by maximum likelihood (ML) methodology. It is based on input data of a `readCatdata` object. Linear, log-linear and functional linear models may be subsequently fitted, respectively, using functions `linML()`, `loglinML()` and `funlinWLS()`.

Usage

```
satMarML(catdataobj, missing="MAR", method="EM", start, zero,
maxit=100, trace=0, epsilon1=1e-6, epsilon2=1e-6, zeroN, digits)
```

Arguments

`catdataobj`	`readCatdata` object.
`missing`	the covariance matrix (based on a Fisher information matrix) of the estimates for the marginal probabilities of categorization may be computed under `"MAR"` (default) or under `"MCAR"` model.
`method`	the iterative processes available are: `"EM"` (Expectation-Maximization), `"FS-MCAR"` (Fisher scoring under MCAR), and `"NR/FS-MAR"` (Fisher scoring under MAR or Newton-Raphson under MAR or MCAR); `"EM"` is the default option, because it is the most stable, although in some cases, the default maximum number of iterations may not be enough due to its slow rate of convergence; as the ML estimates of the marginal probabilities are the same either under MAR or MCAR, one may use the iterative process `"FS-MCAR"` even though one is willing to assume MAR; `"FS-MCAR"` is generally more stable than `"NR/FS-MAR"` when there are sampling zeros, but both iterative processes still may easily jump to a negative estimate and/or generate a singular covariance matrix.
`start`	by default, the function uses the proportions of the complete data as starting values in the iterative process, but the current argument allows the user to inform an alternative starting value for all marginal probabilities except the one corresponding to the last category of each multinomial, i.e., a vector of dimension `S*(R-1)`, where S represents the number of subpopulations and R, the number of response categories.

`zero`	when there are sampling zeros in the complete data, these frequencies are replaced by small values just for the computation of the starting values; this avoids the use of starting values on the boundary of the parameter space and also allows to incorporate information from other missingness patterns in the EM iterative process; by default, the function replaces the values by `1/(R*ns1)`, where *ns1* is the sample size associated to the subpopulation with completely classified data; the user may indicate an alternative vector with *S* values to be used for each subpopulation or an unique value to be used for all subpopulations; the values must be non-negative and less or equal to 0.5.
`maxit`	the maximum number of iterations (the default is 100).
`trace`	the alternatives are: *0* for no printing (default), *1* for showing only the value of the likelihood ratio statistic at each iteration of the iterative process, and *2* for including also the parameter estimates at each iteration.
`epsilon1`	the convergence criterion of the iterative process is attained if the absolute difference of the values of the likelihood ratio statistic of successive iterations is less than the value defined in `epsilon1`, `1e-6` by default.
`epsilon2`	the convergence criterion of the iterative process is attained if the absolute differences of the values of estimates for all parameters of the marginal probabilities of categorization in consecutive iterations are less than the value defined in `epsilon2`, `1e-6` by default.
`zeroN`	values used to replace null frequencies in the denominator of the Neyman statistic; by default, the function replaces the values by `1/(R*nst)`, where *nst* is the sample size of the missingness pattern associated to the corresponding subpopulation; the user may indicate alternative values in a matrix with *S* rows and an additional column relatively to the number of columns of *Rp*; the first column relates to the completely categorized "missingness" patterns, and the remaining columns to the other missingness patterns as they appear in *Rp*; the values must be non-negative and less or equal to 0.5.
`digits`	integer value indicating the number of decimal places to round results when shown by `print` and `summary`; this argument works also when specified directly in both generic functions; default value is 4.

Details

The generic functions `print` and `summary` are used to print the results and to obtain a summary thereof.

Value

An object of the class `satMarML` is a list containing most of the components of the `readCatdata` source object informed in the argument `catdataobj` as well as the following components:

`theta`	vector of ML estimates for all product-multinomial probabilities under the saturated model for the marginal probabilities of categorization and an assumption of an ignorable missingness mechanism; this is the same under MAR and under MCAR.

`Vtheta`	corresponding estimated covariance matrix based on the Fisher information matrix obtained under the assumed missingness mechanism, leading to different results depending whether the assumption is MAR or MCAR).
`QvMCAR`	likelihood ratio statistic for the conditional test of MCAR given a MAR assumption.
`QpMCAR`	Pearson statistic for the conditional test of MCAR given a MAR assumption.
`QnMCAR`	Neyman statistic for the conditional test of MCAR given a MAR assumption.
`glMCAR`	degrees of freedom for the conditional tests of MCAR given a MAR assumption.
`alphast`	ML estimates for the conditional probabilities of missingness under the assumed missingness mechanism (MAR or MCAR).
`yst`	ML estimates for the augmented frequencies under the saturated model for the marginal probabilities and the assumed missingness mechanism (MAR or MCAR).

References

Paulino, C.D. e Singer, J.M. (2006). *Analise de dados categorizados* (in Portuguese). Sao Paulo: Edgard Blucher.

Poleto, F.Z. (2006). *Analise de dados categorizados com omissao* (in Portuguese). Dissertacao de mestrado. IME-USP. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2007). *Analyzing categorical data with complete or missing responses using the Catdata package*. Unpublished vignette. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2012). A product-multinomial framework for categorical data analysis with missing responses. To appear in *Brazilian Journal of Probability and Statistics*. `http://imstat.org/bjps/papers/BJPS198.pdf`.

Singer, J. M., Poleto, F. Z. and Paulino, C. D. (2007). Catdata: software for analysis of categorical data with complete or missing responses. *Actas de la XII Reunion Cientifica del Grupo Argentino de Biometria y I Encuentro Argentino-Chileno de Biometria*. `http://www.poleto.com/SingerPoletoPaulino2007GAB.pdf`.

Examples

```
#Example 13.4 of Paulino and Singer (2006)
e134.TF<-c(12,4,5,2, 50,31, 27,12)
e134.Zp<-cbind(kronecker(diag(2),rep(1,2)),kronecker(rep(1,2),diag(2)))
e134.Rp<-c(2,2)
e134.catdata<-readCatdata(TF=e134.TF,Zp=e134.Zp,Rp=e134.Rp)
e134.satmcarml<-satMarML(e134.catdata,missing="MCAR")
e134.satmarml<-satMarML(e134.catdata,method="FS-MCAR")
e134.satmarml2<-satMarML(e134.catdata,method="NR/FS-MAR")
e134.satmcarml

#Compare the estimates of the probabilities, standard errors,
#number of iterations and augmented frequencies
summary(e134.satmcarml)
summary(e134.satmarml)
summary(e134.satmarml2)
```

```
#Example 1 of Poleto et al (2012)
smoking.TF<-rbind(c(167,17,19,10,1,3,52,10,11, 176,24,121, 28,10,12),
  c(120,22,19, 8,5,1,39,12,12, 103, 3, 80, 31, 8,14))

smoking.Zp <- kronecker(t(rep(1,2)),
cbind(kronecker(diag(3),rep(1,3)),
       kronecker(rep(1,3),diag(3))))

smoking.Rp<-rbind(c(3,3),c(3,3))
smoking.catdata<-readCatdata(TF=smoking.TF,Zp=smoking.Zp,Rp=smoking.Rp)
smoking.catdata

smoking.satmcarml<-satMarML(smoking.catdata,missing="MCAR")
smoking.satmarml<-satMarML(smoking.catdata,method="FS-MCAR")
smoking.satmarml2<-satMarML(smoking.catdata,method="NR/FS-MAR")
smoking.satmarml
summary(smoking.satmcarml)
summary(smoking.satmarml)
summary(smoking.satmarml2)
```

`satMcarWLS` *Fitting Saturated Models for the Marginal Probabilities of Categorization via Weighted Least Squares under MCAR assumption*

Description

`satMcarWLS` fits saturated structural models for the marginal probabilities of categorization under a missing completely at random (MCAR) assumption for the missingness mechanism by weighted least squares (WLS) methodology. It is based on input data of a `readCatdata` object. Functional linear models may be subsequently fitted using function `funlinWLS()`.

Usage

```
satMcarWLS(catdataobj, zeroN)
```

Arguments

`catdataobj`	`readCatdata` object.
`zeroN`	values used to replace null frequencies in the denominator of the Neyman statistic; by default, the function replaces the values by `1/(R*nst)` , where *nst* is the sample size of the missingness pattern associated to the corresponding subpopulation; the user may indicate alternative values in a matrix with *S* rows and an additional column relatively to the number of columns of *Rp*; the first column relates to the completely categorized "missingness" patterns, and the remaining columns to the other missingness patterns as they appear in *Rp*; the values must be non-negative and less or equal to 0.5.

Details

The generic functions `print` and `summary` are used to print the results and to obtain a summary thereof.

Value

An object of the class `satMcarWLS` is a list containing most of the components of the `readCatdata` source object informed in the argument

`catdataobj`

as well as the following components:

`theta`	vector of WLS estimates for all product-multinomial probabilities under the saturated model for the marginal probabilities of categorization and a MCAR assumption.
`Vtheta`	corresponding estimated covariance matrix.
`QnMCAR`	the Neyman statistic for the conditional test of MCAR given a MAR assumption.
`glMCAR`	degrees of freedom for the conditional test of MCAR given a MAR assumption.
`yst`	WLS estimates for the augmented frequencies under the saturated model for the marginal probabilities and the MCAR mechanism.

Author(s)

Frederico Zanqueta Poleto(`<frederico@poleto.com>`)
Julio da Motta Singer (`<jmsinger@ime.usp.br>`)
Carlos Daniel Paulino (`<daniel.paulino@math.ist.utl.pt>`)
with the collaboration of
Fabio Mathias Correa (`<fmcorrea@uesc.br>`)
Enio Galinkin Jelihovschi (`<eniojelihovs@gmail.com>`)

References

Paulino, C.D. e Singer, J.M. (2006). *Analise de dados categorizados* (in Portuguese). Sao Paulo: Edgard Blucher.

Poleto, F.Z. (2006). *Analise de dados categorizados com omissao* (in Portuguese). Dissertacao de mestrado. IME-USP. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2007). *Analyzing categorical data with complete or missing responses using the Catdata package*. Unpublished vignette. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2012). A product-multinomial framework for categorical data analysis with missing responses. To appear in *Brazilian Journal of Probability and Statistics*. `http://imstat.org/bjps/papers/BJPS198.pdf`.

Singer, J. M., Poleto, F. Z. and Paulino, C. D. (2007). Catdata: software for analysis of categorical data with complete or missing responses. *Actas de la XII Reunion Cientifica del Grupo Argentino de Biometria y I Encuentro Argentino-Chileno de Biometria*. `http://www.poleto.com/SingerPoletoPaulino2007GAB.pdf`.

Examples

```
#Example 13.4 of Paulino and Singer (2006)
e134.TF<-c(12,4,5,2, 50,31, 27,12)
e134.Zp<-cbind(kronecker(diag(2),rep(1,2)),kronecker(rep(1,2),diag(2)))
e134.Rp<-c(2,2)
e134.catdata<-readCatdata(TF=e134.TF,Zp=e134.Zp,Rp=e134.Rp)
e134.satmcarwls<-satMcarWLS(e134.catdata)
e134.satmcarwls

#Example 1 of Poleto et al (2012)
smoking.TF<-rbind(c(167,17,19,10,1,3,52,10,11, 176,24,121, 28,10,12),
                  c(120,22,19, 8,5,1,39,12,12, 103, 3, 80, 31, 8,14))
smoking.Zp<-t(rep(1,2))%x%cbind(diag(3)%x%rep(1,3), rep(1,3)%x%diag(3))
smoking.Rp<-rbind(c(3,3),c(3,3))
smoking.catdata<-readCatdata(TF=smoking.TF,Zp=smoking.Zp,Rp=smoking.Rp)
smoking.satmcarwls<-satMcarWLS(smoking.catdata)
smoking.satmcarwls
```

`waldTest` *Wald Test*

Description

`waldTest` allows to perform a Wald test of certain hypotheses on the parameters of models fitted under a freedom equation formulation for both complete and missing data.

Usage

```
waldTest(obj, C, C0)
```

Arguments

`obj`	object of the class `linML`, `loglinML` or `funlinWLS`.
`C`	full rank matrix with number of columns equal to the number of parameters of `obj`.
`C0`	a vector of constants with dimension equal to the number of rows of `C`; by default, it is a vector of zeros.

Details

`waldTest` performs a test for hypotheses of the type H:C%*%Beta=C0, where Beta are the parameters of the model in `obj`. The generic function `print` is used to print the results.

Author(s)

Frederico Zanqueta Poleto(<frederico@poleto.com>)
Julio da Motta Singer (<jmsinger@ime.usp.br>)
Carlos Daniel Paulino (<daniel.paulino@math.ist.utl.pt>)
with the collaboration of
Fabio Mathias Correa (<fmcorrea@uesc.br>)
Enio Galinkin Jelihovschi (<eniojelihovs@gmail.com>)

References

Paulino, C.D. e Singer, J.M. (2006). *Analise de dados categorizados* (in Portuguese). Sao Paulo: Edgard Blucher.

Poleto, F.Z. (2006). *Analise de dados categorizados com omissao* (in Portuguese). Dissertacao de mestrado. IME-USP. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2007). *Analyzing categorical data with complete or missing responses using the Catdata package*. Unpublished vignette. `http://www.poleto.com/missing.html`.

Poleto, F.Z., Singer, J.M. e Paulino, C.D. (2012). A product-multinomial framework for categorical data analysis with missing responses. To appear in *Brazilian Journal of Probability and Statistics*. `http://imstat.org/bjps/papers/BJPS198.pdf`.

Singer, J. M., Poleto, F. Z. and Paulino, C. D. (2007). Catdata: software for analysis of categorical data with complete or missing responses. *Actas de la XII Reunion Cientifica del Grupo Argentino de Biometria y I Encuentro Argentino-Chileno de Biometria*. `http://www.poleto.com/SingerPoletoPaulino2007GAB.pdf`.

Examples

```
#Example 11.7 of Paulino and Singer (2006)
e117.TF<-rbind(c(28,40,68),
               c( 5,21,49),
               c( 1, 4,15))

e117.catdata<-readCatdata(TF=e117.TF)
e117.A<-kronecker(diag(3),cbind(diag(2),rep(-1,2)))
e117.X1<-rbind(c(1,0,0,0),
               c(0,1,0,0),
               c(1,0,2,0),
               c(0,1,1,0),
               c(1,0,0,2),
               c(0,1,0,1))

e117.loglinwls1<-funlinWLS(model=c("lin","log"), obj=e117.catdata,
A1=e117.A,XL=e117.X1)
waldTest(obj=e117.loglinwls1,C=cbind(0*diag(2),diag(2)))
```

Index