Package ‘factormodel’

October 13, 2022

Title  Factor Model Estimation Using Proxy Variables
Version  1.0
License  GPL-3
Encoding  UTF-8
RoxygenNote  7.1.1
Imports  dplyr, nnet, pracma, stats, utils, gtools
Suggests  knitr, rmarkdown
VignetteBuilder  knitr
NeedsCompilation  no
Author  Yujung Hwang [aut, cre](<https://orcid.org/0000-0002-8136-8987>)
Maintainer  Yujung Hwang <yujungghwang@gmail.com>
Repository  CRAN
Date/Publication  2021-06-04 07:40:05 UTC

R topics documented:

cproxyme ................................................................. 2
dproxyme ................................................................. 3
makeDummy ............................................................. 5
weighted.cov .......................................................... 5
weighted.var .......................................................... 6

Index  7
Description

This function estimates a linear factor model using continuous variables. The linear factor model to estimate has the following form. proxy = intercept + factorloading * (latent variable) + measurement error The measurement error is assumed to follow a Normal distribution with a mean zero and a variance, which needs to be estimated.

Usage

cproxyme(dat, anchor = 1, weights = NULL)

Arguments

dat A proxy variable data frame list.
anchor This is a column index of an anchoring proxy variable. Default is 1. That is, the code will use the first column in dat data frame as an anchoring variable.
weights An optional weight vector

Value

Returns a list of 3 components:

alpha0 This is a vector of intercepts in a linear factor model. The k-th entry is the intercept of k-th proxy variable factor model.
alpha1 This is a vector of factor loadings. The k-th entry is the factor loading of k-th proxy variable. The factor loading of anchoring variable is normalized to 1.
varnu This is a vector of variances of measurement errors in proxy variables. The k-th entry is the variance of k-th proxy measurement error. The measurement error is assumed to follow a Normal distribution with mean 0.
mtheta This is a mean of the latent variable. It is equal to the mean of the anchoring proxy variable.
varttheta This is a variance of the latent variable.

Author(s)

Yujung Hwang, <yujungghwang@gmail.com>

References


**dproxyme**

### Examples

```r
dat1 <- data.frame(proxy1=c(1,2,3),proxy2=c(0.1,0.3,0.6),proxy3=c(2,3,5))
cproxyme(dat=dat1,anchor=1)
## you can specify weights
  cproxyme(dat=dat1,anchor=1,weights=c(0.1,0.5,0.4))
```

### Description

This function estimates measurement stochastic matrices of discrete proxy variables.

### Usage

```r
dproxyme(
  dat,  
  sbar = 2, 
  initvar = 1, 
  initvec = NULL, 
  seed = 210313, 
  tol = 0.005, 
  maxiter = 200, 
  miniter = 10, 
  minobs = 100, 
  maxiter2 = 1000, 
  trace = FALSE, 
  weights = NULL
)
```

### Arguments

- **dat** A proxy variable data frame list.
- **sbar** A number of discrete types. Default is 2.
- **initvar** A column index of a proxy variable to initialize the EM algorithm. Default is 1. That is, the proxy variable in the first column of "dat" is used for initialization.
- **initvec** This vector defines how to group the initvar to initialize the EM algorithm.
- **seed** Seed. Default is 210313 (birthday of this package).
- **tol** A tolerance for EM algorithm. Default is 0.005.
- **maxiter** A maximum number of iterations for EM algorithm. Default is 200.
- **miniter** A minimum number of iterations for EM algorithm. Default is 10.
- **minobs** Compute likelihood of a proxy variable only if there are more than "minobs" observations. Default is 100.
maxiter2  Maximum number of iterations for "multinom". Default is 1000.
trace     Whether to trace EM algorithm progress. Default is FALSE.
weights   An optional weight vector

Value

Returns a list of 5 components:

M_param This is a list of estimated measurement (stochastic) matrices. The k-th matrix is a
measurement matrix of a proxy variable saved in the kth column of dat data frame (or matrix).
The ij-th element in a measurement matrix is the conditional probability of observing j-th
(largest) proxy response value conditional on that the latent type is i.
M_param_col This is a list of column labels of 'M_param' matrices
M_param_row This is a list of row labels of 'M_param' matrices. It is simply c(1:sbar).
mparam   This is a list of multinomial logit coefficients which were used to compute 'M_param'
matrices. These coefficients are useful to compute the likelihood of proxy responses.
typeprob  This is a type probability matrix of size N-by-sbar. The ij-th entry of this matrix gives
the probability of observation i to have type j.

Author(s)

Yujung Hwang, <yujungghwang@gmail.com>

References

Dempster, Arthur P., Nan M. Laird, and Donald B. Rubin (1977) "Maximum likelihood from in-
complete data via the EM algorithm." Journal of the Royal Statistical Society: Series B

Hu, Yingyao (2008) Identification and estimation of nonlinear models with misclassification error
doi: 10.1016/j.jeconom.2007.12.001

Hu, Yingyao (2017) The econometrics of unobservables: Applications of measurement error mod-
els in empirical industrial organization and labor economics. Journal of Econometrics, 200(2),

Hwang, Yujung (2021) Identification and Estimation of a Dynamic Discrete Choice Models with


Examples

dat1 <- data.frame(proxy1=c(1,2,3),proxy2=c(2,3,4),proxy3=c(4,3,2))
## default minimum num of obs to run an EM algorithm is 10
dproxyme(dat=dat1,sbar=2,initvar=1,minobs=3)
## you can specify weights
dproxyme(dat=dat1,sbar=2,initvar=1,minobs=3,weights=c(0.1,0.5,0.4))
**makeDummy**

**Description**
This function is to make dummy variables using a discrete variable.

**Usage**
```
makeDummy(tZ)
```

**Arguments**
- **tZ**: An input vector

**Value**
Returns dZ, a matrix of size length(tZ)-by-card(tZ):
The ij-th element in dZ is 1 if tZ[i] is equal to the j-th largest value of tZ. And the ij-th element in DZ is 0 otherwise. The row sum of dZ must be 1 by construction.

**Author(s)**
Yujung Hwang, <yujungghwang@gmail.com>

**Examples**
```
makeDummy(c(1,2,3))
```

---

**weighted.cov**

**Description**
This function is to compute an unbiased sample weighted covariance. The function uses only pairwise complete observations.

**Usage**
```
weighted.cov(x, y, w = NULL)
```

**Arguments**
- **x**: An input vector to compute a covariance, cov(x,y)
- **y**: An input vector to compute a covariance, cov(x,y)
- **w**: A weight vector
**Value**

Returns an unbiased sample weighted covariance

**Author(s)**

Yujung Hwang, <yujungghwang@gmail.com>

**Examples**

```r
# If you do not specify weights, 
# it returns the usual unweighted sample covariance 
weighted.cov(x=c(1,3,5),y=c(2,3,1))

weighted.cov(x=c(1,3,5),y=c(2,3,1),w=c(0.1,0.5,0.4))
```

---

**weighted.var**

**Description**

This function is to compute an unbiased sample weighted variance.

**Usage**

```r
weighted.var(x, w = NULL)
```

**Arguments**

- `x`: A vector to compute a variance, `var(x)`
- `w`: A weight vector

**Value**

Returns an unbiased sample weighted variance

**Author(s)**

Yujung Hwang, <yujungghwang@gmail.com>

**Examples**

```r
## If you do not specify weights, 
## it returns the usual unweighted sample variance 
weighted.var(x=c(1,3,5))

weighted.var(x=c(1,3,5),w=c(0.1,0.5,0.4))
```
Index

cproxyme, 2

dproxyme, 3

makeDummy, 5

weighted.cov, 5
weighted.var, 6