

# Package ‘poLCA’

April 19, 2009

**Version** 1.1

**Date** 2007-11-01

**Title** Polytomous variable Latent Class Analysis

**Author** Drew Linzer <dlinzer@emory.edu>, Jeffrey Lewis <jblewis@ucla.edu>.

**Maintainer** Drew Linzer <dlinzer@emory.edu>

**Depends** scatterplot3d, MASS

**Description** Latent class analysis and latent class regression models for polytomous outcome variables. Also known as latent structure analysis.

**License** GPL (>= 2)

**URL** <http://userwww.service.emory.edu/~dlinzer/poLCA>

**Repository** CRAN

**Date/Publication** 2007-11-03 18:43:29

## R topics documented:

poLCA-package . . . . .	2
carcinoma . . . . .	2
cheating . . . . .	3
election . . . . .	4
gss82 . . . . .	6
poLCA . . . . .	7
poLCA.makeplot.dich . . . . .	11
poLCA.makeplot.poly . . . . .	12
poLCA.reorder . . . . .	13
poLCA.simdata . . . . .	14
print.poLCA . . . . .	17
rmulti . . . . .	18
values . . . . .	19

<b>Index</b>	<b>20</b>
--------------	-----------

---

poLCA-package	<i>Latent class analysis and latent class regression models for polytomous outcome variables</i>
---------------	--

---

### Description

This package estimates latent class and latent class regression models for polytomous outcome variables.

### Details

Package:	poLCA
Type:	Package
Version:	1.1
Date:	2007-11-01
License:	GPL (>= 2)

The function `poLCA` estimates the latent class model.

### Author(s)

Drew Linzer <dlinzer@emory.edu> and Jeffrey Lewis <jblewis@ucla.edu>

Maintainer: Drew Linzer <dlinzer@emory.edu>

Website: <http://userwww.service.emory.edu/~dlinzer/poLCA>

---

carcinoma	<i>Diagnoses of carcinoma (sample data)</i>
-----------	---

---

### Description

Dichotomous ratings by seven pathologists of 118 slides for the presence or absence of carcinoma in the uterine cervix. Pathologists are labeled A through G. There were 20 different observed response patterns. This data set appears in Agresti (2002, p. 542) as Table 13.1.

### Usage

```
data(carcinoma)
```

### Format

A data frame with 118 observations on 7 variables representing pathologist ratings with 1 denoting "no" and 2 denoting "yes".

**Source**

Agresti, Alan. 2002. *Categorical Data Analysis, second edition*. Hoboken: John Wiley & Sons.

**Examples**

```
##
## Replication of latent class models in Agresti (2002, p. 543),
## Table 13.2 and Table 13.3.
##
data(carcinoma)
f <- cbind(A,B,C,D,E,F,G)~1
lca2 <- polCA(f, carcinoma, nclass=2)      # log-likelihood: -317.2568
lca3 <- polCA(f, carcinoma, nclass=3)      # log-likelihood: -293.705
lca4 <- polCA(f, carcinoma, nclass=4, nrep=10, maxiter=5000)      # log-likelihood: -289.2858
```

---

cheating

*GPA and chronic cheating (sample data)*

---

**Description**

Dichotomous responses by 319 undergraduates to four questions about cheating behavior, and each student's academic GPA.

Students responded either (1) no or (2) yes as to whether they had ever lied to avoid taking an exam (LIEEXAM), lied to avoid handing a term paper in on time (LIEPAPER), purchased a term paper to hand in as their own or had obtained a copy of an exam prior to taking the exam (FRAUD), or copied answers during an exam from someone sitting near to them (COPYEXAM).

The GPA variable is partitioned into five groups: (1) 2.99 or less; (2) 3.00-3.25; (3) 3.26-3.50; (4) 3.51-3.75; (5) 3.76-4.00.

This data set appears in Dayton (1998, pp. 33 and 85) as Tables 3.4 and 7.1.

**Usage**

```
data(cheating)
```

**Format**

A data frame with 319 observations on 5 variables. Note: GPA data were not available for four students who reported never cheating.

**Source**

Dayton, C. Mitchell. 1998. *Latent Class Scaling Analysis*. Thousand Oaks, CA: SAGE Publications.

**Examples**

```
##
## Replication of latent class models in Dayton (1998)
##
## Example 1. Two-class LCA. (Table 3.3, p. 32)
##
data(cheating)
f <- cbind(LIEEXAM,LIEPAPER,FRAUD,COPYEXAM)~1
ch2 <- polCA(f,cheating,nclass=2)          # log-likelihood: -440.0271

##
## Example 2. Two-class latent class regression using
## GPA as a covariate to predict class membership as
## "cheaters" vs. "non-cheaters".
## (Table 7.1, p. 85, and Figure 7.1, p. 86)
##
f2 <- cbind(LIEEXAM,LIEPAPER,FRAUD,COPYEXAM)~GPA
ch2c <- polCA(f2,cheating,nclass=2)       # log-likelihood: -429.6384
GPAmat <- cbind(1,c(1:5))
exb <- exp(GPAmat %*% ch2c$coeff)
matplot(c(1:5),cbind(1/(1+exb),exb/(1+exb)),type="l",lwd=2,
        main="GPA as a predictor of persistent cheating",
        xlab="GPA category, low to high",
        ylab="Probability of latent class membership")
text(1.7,0.3,"Cheaters")
text(1.7,0.7,"Non-cheaters")

##
## Compare results from Example 1 to Example 2.
## Non-simultaneous estimation of effect of GPA on latent class
## membership biases the estimated effect in Example 1.
##
cheatcl <- which.min(ch2$P)
predcc <- sapply(c(1:5),function(v) mean(ch2$posterior[cheating$GPA==v,cheatcl],na.rm=TRUE))
## Having run Ex.2, add to plot:
matplot(c(1:5),cbind(1-predcc,predcc),type="l",lwd=2,add=TRUE)
text(4,0.14,"Cheaters\n (non-simul. estimate)")
text(4,0.87,"Non-cheaters\n (non-simul. estimate)")
```

election

*2000 National Election Studies survey (sample data)***Description**

Survey data from the 2000 American National Election Study. Two sets of six questions with four responses each, asking respondents' opinions of how well various traits (moral, caring, knowledgeable, good leader, dishonest, intelligent) describe presidential candidates Al Gore and George W. Bush. The responses are (1) Extremely well; (2) Quite well; (3) Not too well; (4) Not well at all.

The data set also includes potential covariates `VOTE3`, the respondent's 2000 vote choice (when asked); `AGE`, the respondent's age; `EDUC`, the respondent's level of education; `GENDER`, the respondent's gender; and `PARTY`, the respondent's Democratic-Republican partisan identification.

`VOTE3` is coded as (1) Gore; (2) Bush; (3) Other.

`EDUC` is coded as (1) 8 grades or less; (2) 9-11 grades, no further schooling; (3) High school diploma or equivalency; (4) More than 12 years of schooling, no higher degree; (5) Junior or community college level degree; (6) BA level degrees, no advanced degree; (7) Advanced degree.

`GENDER` is coded as (1) Male; (2) Female.

`PARTY` is coded as (1) Strong Democrat; (2) Weak Democrat; (3) Independent-Democrat; (4) Independent-Independent; (5) Independent-Republican; (6) Weak Republican; (7) Strong Republican.

## Usage

```
data(election)
```

## Format

A data frame with 1311 observations on 17 survey variables.

## Source

The National Election Studies (<http://www.electionstudies.org>). THE 2000 NATIONAL ELECTION STUDY [dataset]. Ann Arbor, MI: University of Michigan, Center for Political Studies [producer and distributor].

## Examples

```
##
## 2000 American National Election Study analysis
##
## Example 1. Latent class models with one (loglinear independence) to three classes.
##
data(election)
f <- cbind(MORALG, CARESG, KNOWG, LEADG, DISHONG, INTELG, MORALB, CARESB, KNOWB, LEADB, DISHONB, INTELB)
nes1 <- polCA(f, election, nclass=1)      # log-likelihood: -18647.31
nes2 <- polCA(f, election, nclass=2)     # log-likelihood: -17344.92
nes3 <- polCA(f, election, nclass=3)     # log-likelihood: -16714.66

##
## Example 2. Three-class model with a single covariate (party, age).
##
f2a <- cbind(MORALG, CARESG, KNOWG, LEADG, DISHONG, INTELG, MORALB, CARESB, KNOWB, LEADB, DISHONB, INTELB)
nes2a <- polCA(f2a, election, nclass=3, nrep=5) # log-likelihood: -16222.32
pidmat <- cbind(1, c(1:7))
exb <- exp(pidmat %*% nes2a$coeff)
matplot(c(1:7), (cbind(1, exb) / (1+rowSums(exb))), ylim=c(0, 1), type="l",
        main="Party ID as a predictor of candidate affinity class",
        xlab="Party ID: strong Democratic (1) to strong Republican (7)",
        ylab="Probability of latent class membership", lwd=2, col=1)
```

```

text(5.9,0.35,"Other")
text(5.4,0.7,"Bush affinity")
text(1.8,0.6,"Gore affinity")

f2b <- cbind(MORALG,CARESG,KNOWG,LEADG,DISHONG,INTELG,MORALB,CARESB,KNOWB,LEADB,DISHONB,INTELG)
nes2b <- polCA(f2b,election,nclass=3,nrep=5) # log-likelihood: -16625.96
agemat <- cbind(1,c(18:90))
exb <- exp(agemat %*% nes2b$coeff)
matplot(c(18:90),(cbind(1,exb)/(1+rowSums(exb))),ylim=c(0,1),type="l",
        main="Age as a predictor of candidate affinity class",
        xlab="Age",
        ylab="Probability of latent class membership",lwd=2,col=1)
text(30,0.55,"Other")
text(30,0.33,"Bush affinity")
text(30,0.17,"Gore affinity")

##
## Example 3. Three-class model with covariates
## age, education, and age*education interaction.
## Graph shows predicted class probabilities by age,
## for high-school grads versus college grads.
##
f3 <- cbind(MORALG,CARESG,KNOWG,LEADG,DISHONG,INTELG,MORALB,CARESB,KNOWB,LEADB,DISHONB,INTELG)
nes3cov <- polCA(f3,election,nclass=3,nrep=5) # log-likelihood: -16601.04
predmat <- list()
for (i in 1:3) {
  predmat[[i]] <- matrix(NA,nrow=73,ncol=2)
  ivmat.HS <- cbind(1,c(18:90),3,(c(18:90)*3))
  ivmat.COLL <- cbind(1,c(18:90),6,(c(18:90)*6))
  exb.HS <- exp(ivmat.HS %*% nes3cov$coeff)
  exb.COLL <- exp(ivmat.COLL %*% nes3cov$coeff)
  predmat[[i]][,1] <- (cbind(1,exb.HS)/(1+rowSums(exb.HS)))[,i]
  predmat[[i]][,2] <- (cbind(1,exb.COLL)/(1+rowSums(exb.COLL)))[,i]
}
matplot(c(18:90),predmat[[1]],ylim=c(0.1,0.6),type="l",lty=c(1,2),col="blue",
        main="Age and Education as predictors of candidate affinity class",
        xlab="Age",
        ylab="Probability of latent class membership",lwd=2)
matplot(c(18:90),predmat[[2]],type="l",lty=c(1,2),col="red",lwd=2,add=TRUE)
matplot(c(18:90),predmat[[3]],type="l",lty=c(1,2),col="limegreen",lwd=2,add=TRUE)
text(27,0.55,"HS: Other"); text(23,0.49,"College:\n Other")
text(27,0.32,"HS: Bush affinity"); text(31,0.27,"College:\n Bush affinity")
text(27,0.15,"HS: Gore affinity"); text(22,0.235,"College:\n Gore affinity")

```

## Description

Attitudes towards survey taking across two dichotomous and two trichotomous items among 1202 white respondents to the 1982 General Social Survey. Respondents give their opinion of the pur-

pose of surveys (PURPOSE; good/depends/waste of time and money), the accuracy of surveys (ACCURACY; mostly true/not true), their understanding of survey questions (UNDERSTA; good/fair, poor), and how well they cooperated with the interviewer (COOPERAT; interested/cooperative/impatient, hostile). This data set appears in McCutcheon (1987, p. 30) as Table 3.1.

### Usage

```
data(gss82)
```

### Format

A data frame with 1202 observations on 4 survey variables.

### Source

McCutcheon, A.L. 1987. *Latent class analysis*. Newbury Park: SAGE Publications.

### Examples

```
data(gss82)
f <- cbind(PURPOSE, ACCURACY, UNDERSTA, COOPERAT) ~ 1
gss.lc2 <- poLCA(f, gss82, nclass=2) # log-likelihood = -2783.268
gss.lc3 <- poLCA(f, gss82, nclass=3, maxiter=3000, nrep=10) # log-likelihood = -2754.545
gss.lc4 <- poLCA(f, gss82, nclass=4, maxiter=15000, nrep=10, tol=1e-7) # log-likelihood = -2746.6
```

---

poLCA

*Latent class analysis of polytomous outcome variables*

---

### Description

Estimates latent class and latent class regression models for polytomous outcome variables.

### Usage

```
poLCA(formula, data, nclass = 2, maxiter = 1000, graphs = FALSE,
      tol = 1e-10, na.rm = TRUE, probs.start = NULL, nrep = 1,
      verbose = TRUE)
```

### Arguments

formula	A formula expression of the form <code>response ~ predictors</code> . The details of model specification are given below.
data	A data frame containing variables in <code>formula</code> . Manifest variables must contain <i>only</i> integer values, and must be coded with consecutive values from 1 to the maximum number of outcomes for each variable. All missing values should be entered as NA.
nclass	The number of latent classes to assume in the model. Setting <code>nclass=1</code> results in poLCA estimating the loglinear independence model. The default is two.

<code>maxiter</code>	The maximum number of iterations through which the estimation algorithm will cycle.
<code>graphs</code>	Logical, for whether <code>poLCA</code> should graphically display the parameter estimates at each stage of the updating algorithm. The default is <code>FALSE</code> .
<code>tol</code>	A tolerance value for judging when convergence has been reached. When the one-iteration change in the estimated log-likelihood is less than <code>tol</code> , the estimation algorithm stops updating and considers the maximum log-likelihood to have been found.
<code>na.rm</code>	Logical, for how <code>poLCA</code> handles cases with missing values on the manifest variables. If <code>TRUE</code> , those cases are removed (listwise deleted) before estimating the model. If <code>FALSE</code> , cases with missing values are retained. Cases with missing covariates are always removed. The default is <code>TRUE</code> .
<code>probs.start</code>	A list of matrices of class-conditional response probabilities to be used as the starting values for the estimation algorithm. Each matrix in the list corresponds to one manifest variable, with one row for each latent class, and one column for each outcome. The default is <code>NULL</code> , producing random starting values. Note that if <code>nrep&gt;1</code> , then any user-specified <code>probs.start</code> values are only used in the first of the <code>nrep</code> attempts.
<code>nrep</code>	Number of times to estimate the model, using different values of <code>probs.start</code> . The default is one. Setting <code>nrep&gt;1</code> automates the search for the global—rather than just a local—maximum of the log-likelihood function. <code>poLCA</code> returns the parameter estimates corresponding to the model with the greatest log-likelihood.
<code>verbose</code>	Logical, indicating whether <code>poLCA</code> should output to the screen the results of the model. If <code>FALSE</code> , no output is produced. The default is <code>TRUE</code> .

## Details

Latent class analysis, also known as latent structure analysis, is a technique for the analysis of clustering among observations in multi-way tables of qualitative/categorical variables. The central idea is to fit a model in which any confounding between the manifest variables can be explained by a single unobserved "latent" categorical variable. `poLCA` uses the assumption of local independence to estimate a mixture model of latent multi-way tables, the number of which (`nclass`) is specified by the user. Estimated parameters include the class-conditional response probabilities for each manifest variable, the "mixing" proportions denoting population share of observations corresponding to each latent multi-way table, and coefficients on any class-predictor covariates, if specified in the model.

Model specification: Latent class models have more than one manifest variable, so the response variables are `cbind(dv1, dv2, dv3...)` where `dv#` refer to variable names in the data frame. For models with no covariates, the formula is `cbind(dv1, dv2, dv3)~1`. For models with covariates, replace the `~1` with the desired function of predictors `iv1, iv2, iv3...` as, for example, `cbind(dv1, dv2, dv3)~iv1+iv2*iv3`.

`poLCA` treats all manifest variables as qualitative/categorical/nominal – NOT as ordinal.

## Value

`poLCA` returns an object of class `poLCA`; a list containing the following elements:

<code>y</code>	data frame of manifest variables.
<code>x</code>	data frame of covariates, if specified.
<code>N</code>	number of cases used in model.
<code>Nobs</code>	number of fully observed cases (less than or equal to <code>N</code> ).
<code>probs</code>	estimated class-conditional response probabilities.
<code>probs.se</code>	standard errors of estimated class-conditional response probabilities, in the same format as <code>probs</code> .
<code>P</code>	sizes of each latent class; equal to the mixing proportions in the basic latent class model, or the mean of the priors in the latent class regression model.
<code>P.se</code>	the standard errors of the estimated <code>P</code> .
<code>posterior</code>	matrix of posterior class membership probabilities.
<code>predclass</code>	vector of predicted class memberships, by modal assignment.
<code>predcell</code>	table of observed vs. predicted cell counts.
<code>llik</code>	maximum value of the log-likelihood.
<code>numiter</code>	number of iterations until reaching convergence.
<code>maxiter</code>	maximum number of iterations through which the estimation algorithm was set to run.
<code>coeff</code>	multinomial logit coefficient estimates on covariates (when estimated). <code>coeff</code> is a matrix with <code>nclass-1</code> columns, and one row for each covariate. All logit coefficients are calculated for classes with respect to class 1.
<code>coeff.se</code>	standard errors of coefficient estimates on covariates (when estimated), in the same format as <code>coeff</code> .
<code>coeff.V</code>	covariance matrix of coefficient estimates on covariates (when estimated).
<code>aic</code>	Akaike Information Criterion.
<code>bic</code>	Bayesian Information Criterion.
<code>Gsq</code>	Likelihood ratio/deviance statistic.
<code>Chisq</code>	Pearson Chi-square goodness of fit statistic for fitted vs. observed multiway tables.
<code>time</code>	length of time it took to run the model.
<code>npar</code>	number of degrees of freedom used by the model (estimated parameters).
<code>resid.df</code>	number of residual degrees of freedom.
<code>attempts</code>	a vector containing the maximum log-likelihood values found in each of the <code>nrep</code> attempts to fit the model.
<code>eflag</code>	Logical, error flag. TRUE if estimation algorithm needed to automatically restart with new initial parameters. A restart is caused in the event of computational/rounding errors that result in nonsensical parameter estimates.
<code>probs.start</code>	A list of matrices containing the class-conditional response probabilities used as starting values in the estimation algorithm. If the algorithm needed to restart (see <code>eflag</code> ), then this contains the starting values used for the final, successful, run.
<code>probs.start.ok</code>	Logical. FALSE if <code>probs.start</code> was incorrectly specified by the user, otherwise TRUE.

**Note**

poLCA uses EM and Newton-Raphson algorithms to maximize the latent class model log-likelihood function. Depending on the starting parameters, this algorithm may only locate a local, rather than global, maximum. This becomes more and more of a problem as `nclass` increases. It is therefore highly advisable to run poLCA multiple times until you are relatively certain that you have located the global maximum log-likelihood. As long as `probs.start=NULL`, each function call will use different (random) initial starting parameters. Alternately, setting `nrep` to a value greater than one enables the user to estimate the latent class model multiple times with a single call to poLCA, thus conducting the search for the global maximizer automatically.

The term "Latent class regression" (LCR) can have two meanings. In this package, LCR models refer to latent class models in which the probability of class membership is predicted by one or more covariates. However, in other contexts, LCR is also used to refer to regression models in which the manifest variable is partitioned into some specified number of latent classes as part of estimating the regression model. It is a way to simultaneously fit more than one regression to the data when the latent data partition is unknown. The `regmix` function in package **fpc** will estimate this other type of LCR model. Because of these terminology issues, the LCR models this package estimates are sometimes termed "latent class models with covariates" or "concomitant-variable latent class analysis," both of which are accurate descriptions of this model.

A more detailed user's manual is available online at <http://userwww.service.emory.edu/~dlinzer/poLCA>.

**References**

- Agresti, Alan. 2002. *Categorical Data Analysis, second edition*. Hoboken: John Wiley & Sons.
- Bandeen-Roche, Karen, Diana L. Miglioretti, Scott L. Zeger, and Paul J. Rathouz. 1997. "Latent Variable Regression for Multiple Discrete Outcomes." *Journal of the American Statistical Association*. 92(440): 1375-1386.
- Hagenaars, Jacques A. and Allan L. McCutcheon, eds. 2002. *Applied Latent Class Analysis*. Cambridge: Cambridge University Press.
- McLachlan, Geoffrey J. and Thriyambakam Krishnan. 1997. *The EM Algorithm and Extensions*. New York: John Wiley & Sons.

**See Also**

`poLCA.simdata`, `lca`, `regmix`, `gllm`

**Examples**

```
##
## Three models without covariates:
## M0: Loglinear independence model.
## M1: Two-class latent class model.
## M2: Three-class latent class model.
##
data(values)
f <- cbind(A,B,C,D)~1
M0 <- poLCA(f,values,nclass=1)           # log-likelihood: -543.6498
M1 <- poLCA(f,values,nclass=2)           # log-likelihood: -504.4677
```

```

M2 <- poLCA(f,values,nclass=3,maxiter=8000)      # log-likelihood: -503.3011

##
## Three-class model with a single covariate.
##
data(election)
f2a <- cbind(MORALG,CARESG,KNOWG,LEADG,DISHONG,INTELG,MORALB,CARESB,KNOWB,LEADB,DISHONB,INTE
nes2a <- poLCA(f2a,election,nclass=3,nrep=5)      # log-likelihood: -16222.32
pidmat <- cbind(1,c(1:7))
exb <- exp(pidmat %**% nes2a$coeff)
matplot(c(1:7),(cbind(1,exb)/(1+rowSums(exb))),ylim=c(0,1),type="l",
        main="Party ID as a predictor of candidate affinity class",
        xlab="Party ID: strong Democratic (1) to strong Republican (7)",
        ylab="Probability of latent class membership",lwd=2,col=1)
text(5.9,0.35,"Other")
text(5.4,0.7,"Bush affinity")
text(1.8,0.6,"Gore affinity")

##
## Create a sample dataset with missing values and estimate the
## latent class model including and excluding the missing values.
## Then plot the estimated class-conditional outcome response
## probabilities against each other for the two methods.
##
simdat3 <- poLCA.simdata(N=8000,niv=2,ndv=5,nclass=3,b=matrix(c(-1,2,-3,1,-2,2),3,2),missval
f3 <- cbind(Y1,Y2,Y3,Y4,Y5)~X1+X2
lc3.miss <- poLCA(f3,simdat3$dat,nclass=3,verbose=FALSE)
probs.start.new <- poLCA.reorder(lc3.miss$probs.start,order(lc3.miss$P))
lc3.miss <- poLCA(f3,simdat3$dat,nclass=3,probs.start=probs.start.new)

lc3.nomiss <- poLCA(f3,simdat3$dat,nclass=3,verbose=FALSE,na.rm=FALSE)
probs.start.new <- poLCA.reorder(lc3.nomiss$probs.start,order(lc3.nomiss$P))
lc3.nomiss <- poLCA(f3,simdat3$dat,nclass=3,na.rm=FALSE,probs.start=probs.start.new)

plot(lc3.miss$probs[[1]],lc3.nomiss$probs[[1]],xlim=c(0,1),ylim=c(0,1),
     xlab="Conditional response probabilities (missing values dropped)",
     ylab="Conditional response probabilities (missing values included)")
for (i in 2:5) { points(lc3.miss$probs[[i]],lc3.nomiss$probs[[i]]) }
abline(0,1,lty=3)

```

---

poLCA.makeplot.dich

*Plot multi-item dichotomous response probabilities*

---

## Description

This function plots estimated response probabilities to a series of dichotomous outcome variables, conditional on estimated class membership, for an arbitrary number of classes. It also shows the estimated population proportion in each class.

**Usage**

```
poLCA.makeplot.dich(probs, P, y, ti)
```

**Arguments**

probs	a list of length equal to the number of outcome variables, containing estimated conditional item response probabilities for each question and class.
P	class membership proportions.
y	matrix of observed outcome variables.
ti	title of upper plot.

**Details**

This function is designed to be called internally from `poLCA` only when the attribute `graphs` is set to `TRUE`, *and* only when every outcome item is dichotomous. If one or more dependent variables are polytomous, then the `poLCA` function will instead call `poLCA.makeplot.poly` to plot the relevant information.

**Note**

This function is simply an elaborate invocation of the `scatterplot3d` command in the **scatterplot3d** package.

**See Also**

`poLCA`, `poLCA.makeplot.poly`

---

```
poLCA.makeplot.poly
```

*Plot multi-item polytomous response probabilities*

---

**Description**

This function plots estimated response probabilities to a series of polytomous outcome variables, conditional on estimated class membership.

**Usage**

```
poLCA.makeplot.poly(probs, r, y, K.j, ti)
```

**Arguments**

probs	a list of length equal to the number of outcome variables, containing estimated conditional item response probabilities for each question and class.
r	class for which to plot response probabilities.
y	matrix of observed outcome variables.
K.j	vector containing number of response items for each dependent variable.
ti	title of probability plots.

**Details**

This function is designed to be called internally from `poLCA` only when the attribute `graphs` is set to `TRUE`, and only if at least one outcome item is polytomous. If all dependent variables are dichotomous, then the `poLCA` function will instead call `poLCA.makeplot.dich` to plot the relevant information.

**Note**

This function is simply an elaborate invocation of the `scatterplot3d` command in the **scatterplot3d** package.

**See Also**

`poLCA`, `poLCA.makeplot.dich`

---

<code>poLCA.reorder</code>	<i>Reorder latent classes in poLCA</i>
----------------------------	--

---

**Description**

A helper function to simplify the reordering of latent classes returned by `poLCA`.

**Usage**

```
poLCA.reorder(probs, o.new)
```

**Arguments**

<code>probs</code>	a list of class-conditional response probabilities previously used as start values to estimate a particular latent class model using <code>poLCA</code> .
<code>o.new</code>	a vector of length equal to the number of latent classes in <code>probs</code> , giving the desired reordering of the latent classes.

**Details**

Because the latent classes outputted by `poLCA` are unordered categories, the numerical order of the classes is arbitrary, and is determined solely by the initial values of the EM algorithm. If `probs.start` is set to `NULL` (the default) when calling `poLCA`, then the function generates the starting values randomly in each run, typically rearranging the latent class labels. The `poLCA.reorder` function is a convenient way to manually adjust the order of the latent classes, by changing the order of the `probs.start`. Refitting the latent class model using these reordered start values will produce a model having the desired category labels.

**Value**

The function returns a list of matrices containing the rearranged (by row) class-conditional response probabilities.

**See Also**[poLCA](#)**Examples**

```
##
## Create sample data and arrange latent classes in order of increasing size.
##
sim <- poLCA.simdata(N=5000,nclass=3,ndv=5,clasdist=c(0.15,0.30,0.55))
f <- cbind(Y1,Y2,Y3,Y4,Y5)~1
lc <- poLCA(f,sim$dat,nclass=3)
probs.start.new <- poLCA.reorder(lc$probs.start,order(lc$P))
lc2 <- poLCA(f,sim$dat,nclass=3,probs.start=probs.start.new)

##
## Using the "cheating" sample data set, make the larger
## non-cheater class the first ("reference") class in a
## latent class regression model. The coefficient on GPA
## now maintains a consistent interpretation.
##
data(cheating)
f2 <- cbind(LIEEXAM,LIEPAPER,FRAUD,COPYEXAM)~GPA
lc.ch <- poLCA(f2,cheating,nclass=2,verbose=FALSE)
probs.start.new <- poLCA.reorder(lc.ch$probs.start,order(lc.ch$P,decreasing=TRUE))
lc.ch <- poLCA(f2,cheating,nclass=2,probs.start=probs.start.new)
```

---

poLCA.simdata

---

*Create simulated cross-classification data*


---

**Description**

Uses the latent class model's assumed data-generating process to create a simulated dataset that can be used to test the properties of the poLCA latent class and latent class regression estimator.

**Usage**

```
poLCA.simdata(N = 5000, probs = NULL, nclass = 2, ndv = 4,
             nresp = NULL, x = NULL, niv = 0, b = NULL,
             clasdist = NULL, missval = FALSE, pctmiss = NULL)
```

**Arguments**

**N** number of observations.

**probs** a list of matrices of dimension `nclass` by `nresp` with each matrix corresponding to one manifest variable, and each row containing the class-conditional outcome probabilities (which must sum to 1) If `probs` is `NULL` (default) then the outcome probabilities are generated randomly.

<code>nclass</code>	number of latent classes. If <code>probs</code> is specified, then <code>nclass</code> is set equal to the number of rows in each matrix in that list. If <code>classdist</code> is specified, then <code>nclass</code> is set equal to the length of that vector. If <code>b</code> is specified, then <code>nclass</code> is set equal to one greater than the number of columns in <code>b</code> . Otherwise, the default is two.
<code>ndv</code>	number of manifest variables. If <code>probs</code> is specified, then <code>ndv</code> is set equal to the number of matrices in that list. If <code>nresp</code> is specified, then <code>ndv</code> is set equal to the length of that vector. Otherwise, the default is four.
<code>nresp</code>	number of possible outcomes for each manifest variable. If <code>probs</code> is specified, then <code>ndv</code> is set equal to the number of columns in each matrix in that list. If both <code>probs</code> and <code>nresp</code> are NULL (default), then the manifest variables are assigned a random number of outcomes between two and five.
<code>x</code>	a matrix of concomitant variables with <code>N</code> rows and <code>niv</code> columns. If <code>x=NULL</code> (default), but <code>niv&gt;0</code> , then <code>niv</code> concomitant variables will be generated as mutually independent random draws from a standard normal distribution.
<code>niv</code>	number of concomitant variables (covariates). Setting <code>niv=0</code> (default) creates a data set assuming no covariates. If <code>nclass=1</code> then <code>niv</code> is automatically set equal to 0. If both <code>x</code> and <code>niv</code> are entered, then the number of columns in <code>x</code> overrides the value of <code>niv</code> . The number of rows in <code>b</code> , less one, also overrides <code>niv</code> .
<code>b</code>	when using covariates, an <code>niv+1</code> by <code>nclass-1</code> matrix of (multinomial) logit coefficients. If <code>b</code> is NULL (default), then coefficients are generated as random integers between -2 and 2.
<code>classdist</code>	a vector of mixing proportions (class population shares) of length <code>nclass</code> . <code>classdist</code> must sum to 1. Disregarded if <code>b</code> is specified or <code>niv&gt;1</code> because then <code>classdist</code> is, in part, a function of the concomitant variables. If <code>classdist</code> is NULL (default), then the mixing proportions are generated randomly.
<code>missval</code>	logical. If TRUE then a fraction <code>pctmiss</code> of the manifest variables are randomly dropped as missing values. Default is FALSE.
<code>pctmiss</code>	percentage of values to be dropped as missing, if <code>missval=TRUE</code> . If <code>pctmiss</code> is NULL (default), then a value between 5 and 40 percent is chosen randomly.

### Details

Note that entering `probs` overrides `nclass`, `ndv`, and `nresp`. It also overrides `classdist` if the length of the `classdist` vector is not equal to the length of the `probs` list. Likewise, if `probs=NULL`, then `length(nresp)` overrides `ndv` and `length(classdist)` overrides `nclass`. Setting `niv>1` causes any user-entered value of `classdist` to be disregarded.

### Value

<code>dat</code>	a data frame containing the simulated variables. Variable names for manifest variables are Y1, Y2, etc. Variable names for concomitant variables are X1, X2, etc.
<code>probs</code>	a list of matrices of dimension <code>nclass</code> by <code>nresp</code> containing the class-conditional response probabilities.

nresp            a vector containing the number of possible outcomes for each manifest variable.  
 b                coefficients on covariates, if used.  
 classdist       mixing proportions corresponding to each latent class.  
 pctmiss         percent of observations missing.  
 trueclass       N by 1 vector containing the "true" class membership for each individual.

## See Also

[poLCA](#)

## Examples

```
##
## Create a sample data set with 3 classes and no covariates
## and run poLCA to recover the specified parameters.
##
probs <- list(matrix(c(0.6,0.1,0.3,      0.6,0.3,0.1,      0.3,0.1,0.6      ),ncol=3,byrow=TRUE)
              matrix(c(0.2,0.8,        0.7,0.3,          0.3,0.7          ),ncol=2,byrow=TRUE)
              matrix(c(0.3,0.6,0.1,    0.1,0.3,0.6,      0.3,0.6,0.1      ),ncol=3,byrow=TRUE)
              matrix(c(0.1,0.1,0.5,0.3, 0.5,0.3,0.1,0.1, 0.3,0.1,0.1,0.5),ncol=4,byrow=TRUE)
              matrix(c(0.1,0.1,0.8,    0.1,0.8,0.1,      0.8,0.1,0.1      ),ncol=3,byrow=TRUE)
simdat <- poLCA.simdata(N=10000,probs,classdist=c(0.2,0.3,0.5))
f1 <- cbind(Y1,Y2,Y3,Y4,Y5)~1
lc1 <- poLCA(f1,simdat$dat,nclass=3)
print(table(lc1$predclass,simdat>trueclass))

##
## Create a sample dataset with 2 classes and three covariates.
## Then compare predicted class memberships when the model is
## estimated "correctly" with covariates to when it is estimated
## "incorrectly" without covariates.
##
simdat2 <- poLCA.simdata(N=5000,ndv=7,niv=3,nclass=2,b=matrix(c(1,-2,1,-1)))
f2a <- cbind(Y1,Y2,Y3,Y4,Y5,Y6,Y7)~X1+X2+X3
lc2a <- poLCA(f2a,simdat2$dat,nclass=2)
f2b <- cbind(Y1,Y2,Y3,Y4,Y5,Y6,Y7)~1
lc2b <- poLCA(f2b,simdat2$dat,nclass=2)
print(table(lc2a$predclass,lc2b$predclass))

##
## Create a sample dataset with missing values and estimate the
## latent class model including and excluding the missing values.
## Then plot the estimated class-conditional outcome response
## probabilities against each other for the two methods.
##
simdat3 <- poLCA.simdata(N=8000,niv=2,ndv=5,nclass=3,b=matrix(c(-1,2,-3,1,-2,2),3,2),missval
f3 <- cbind(Y1,Y2,Y3,Y4,Y5)~X1+X2
lc3.miss <- poLCA(f3,simdat3$dat,nclass=3,verbose=FALSE)
probs.start.new <- poLCA.reorder(lc3.miss$probs.start,order(lc3.miss$P))
lc3.miss <- poLCA(f3,simdat3$dat,nclass=3,probs.start=probs.start.new)
```

```
lc3.nomiss <- poLCA(f3, simdat3$dat, nclass=3, verbose=FALSE, na.rm=FALSE)
probs.start.new <- poLCA.reorder(lc3.nomiss$probs.start, order(lc3.nomiss$P))
lc3.nomiss <- poLCA(f3, simdat3$dat, nclass=3, na.rm=FALSE, probs.start=probs.start.new)

plot(lc3.miss$probs[[1]], lc3.nomiss$probs[[1]], xlim=c(0,1), ylim=c(0,1),
     xlab="Conditional response probabilities (missing values dropped)",
     ylab="Conditional response probabilities (missing values included)")
for (i in 2:5) { points(lc3.miss$probs[[i]], lc3.nomiss$probs[[i]]) }
abline(0,1, lty=3)
```

---

print.poLCA

*Output an object of class poLCA*

---

## Description

Produces nice output of an object of class poLCA as generated by [poLCA](#).

## Usage

```
print.poLCA(x, ...)
```

## Arguments

x                    An object of class poLCA.

...

## See Also

[poLCA](#)

## Examples

```
data(values)
f <- cbind(A,B,C,D)~1
M1 <- poLCA(f, values, nclass=2, verbose=FALSE)
print(M1)
```

---

`rmulti`*Random draws from a multinomial distribution*

---

**Description**

One random draw from a multinomial distribution or list of multinomial distributions.

**Usage**

```
rmulti(p)
```

**Arguments**

`p` matrix of dimension `n` by `r` containing probabilities, for each row, of drawing each of `r` outcomes. `p` may also be entered as a vector, in which case `rmulti` treats it as a matrix of dimension `n=1` by `r`.

**Value**

Returns a vector of length `n`. Each item represents one draw from the multinomial distribution parameterized by the outcome probabilities in each row of `p`.

**Note**

Each row of matrix `p` must sum to 1 or `rmulti` will not work properly.

**Examples**

```
##
## One draw from a three-category multinomial distribution.
##
p1 <- c(0.7,0.2,0.1)
rmulti(p1)

##
## 10,000 draws from a three-category multinomial distribution.
##
n <- 10000
p2 <- matrix(p1,nrow=n,ncol=length(p1),byrow=TRUE)
rmdraws <- rmulti(p2)
table(rmdraws)/n      # should be approximately 0.7, 0.2, 0.1

##
## 10,000 draws from a mixture of three groups of a
## four-category multinomial distribution.
##
group.p <- matrix(c(0.5,0.3,0.2),nrow=n,ncol=3,byrow=TRUE)
group <- rmulti(group.p)
p3 <- t(matrix(NA,nrow=n,ncol=4))
```

```

p3[,group==1] <- c(0.7,0.1,0.1,0.1)
p3[,group==2] <- c(0.1,0.7,0.1,0.1)
p3[,group==3] <- c(0.1,0.1,0.1,0.7)
p3 <- t(p3)
rmdraws3 <- rmulti(p3)
print(table(group,rmdraws3))
print(table(group,rmdraws3)/rowSums(table(group,rmdraws3)))

```

---

values

*Universalistic vs. particularistic values (sample data)*


---

### Description

Dichotomous survey responses from 216 respondents to four questions (A, B, C, D) measuring tendencies towards "universalistic" or "particularistic" values. This data set appears in Goodman (2002, p. 14) as Table 4, and previously appeared in Goodman (1974) and Stouffer and Toby (1951).

### Usage

```
data(values)
```

### Format

A data frame with 216 observations on 4 variables representing survey responses to dichotomous questions, with 1 denoting the "particularistic" values response and 2 denoting the "universalistic" values response.

### Source

Stouffer, S.A. and J. Toby. 1951. "Role conflict and personality." *American Journal of Sociology*. 56: 395:406.

Goodman, Leo A. 1974. "Exploratory Latent-Structure Analysis Using Both Identifiable and Unidentifiable Models." *Biometrika*. 61(2): 215-231.

Goodman, Leo A. 2002. "Latent Class Analysis; The Empirical Study of Latent Types, Latent Variables, and Latent Structures." in Jacques A. Hageaars and Allan L. McCutcheon, eds. *Applied Latent Class Analysis*. Cambridge: Cambridge University Press.

### Examples

```

##
## Replication of latent class models in Goodman (2002),
## Tables 5b, 5c, and 6.
##
data(values)
f <- cbind(A,B,C,D)~1
M0 <- polCA(f,values,nclass=1)           # log-likelihood: -543.6498
M1 <- polCA(f,values,nclass=2)           # log-likelihood: -504.4677
M2 <- polCA(f,values,nclass=3,maxiter=8000) # log-likelihood: -503.3011

```

# Index

## \*Topic **datasets**

- carcinoma, [2](#)
- cheating, [3](#)
- election, [4](#)
- gss82, [6](#)
- values, [18](#)

## \*Topic **documentation**

- poLCA-package, [1](#)

## \*Topic **hplot**

- poLCA.makeplot.dich, [11](#)
- poLCA.makeplot.poly, [12](#)

## \*Topic **methods**

- poLCA, [7](#)
- poLCA.reorder, [13](#)
- poLCA.simdata, [14](#)
- rmulti, [17](#)

## \*Topic **print**

- print.poLCA, [17](#)

carcinoma, [2](#)

cheating, [3](#)

election, [4](#)

gllm, [10](#)

gss82, [6](#)

lca, [10](#)

poLCA, [2](#), [7](#), [12](#), [13](#), [16](#), [17](#)

poLCA-package, [1](#)

poLCA.makeplot.dich, [11](#), [12](#), [13](#)

poLCA.makeplot.poly, [12](#), [12](#)

poLCA.reorder, [13](#)

poLCA.simdata, [10](#), [14](#)

print.poLCA, [17](#)

regmix, [10](#)

rmulti, [17](#)

scatterplot3d, [12](#), [13](#)

values, [18](#)