poLCA:
Polytomous Variable
Latent Class Analysis

Drew A. Linzer
dlinzer@ucla.edu

Jeffrey Lewis
jblewis@ucla.edu

Department of Political Science
University of California, Los Angeles

Website: http://dlinzer.bol.ucla.edu/poLCA

Abstract

poLCA is a software package for the estimation of latent class and latent class regression models for polytomous outcome variables, implemented in the R statistical computing environment. Both models can be called using a single simple command line. The basic latent class model is a finite mixture model in which the component distributions are assumed to be multi-way cross-classification tables with all variables mutually independent. The latent class regression model further enables the researcher to estimate the effects of covariates on predicting latent class membership. poLCA uses expectation-maximization and Newton-Raphson algorithms to find maximum likelihood estimates of the model parameters.

1 Quick Start

This section is provided for users who wish to skip the technical details and proceed directly to the estimation of latent class and latent class regression models.

1.1 Installation

Download the current version of the poLCA software package from the poLCA website or from the Comprehensive R Archive Network (CRAN) by loading R and selecting `Packages > Install package(s)...` from the drop-down menu. Select a nearby CRAN mirror and click `OK`. Scroll down to the poLCA package and click `OK`. The package will automatically download to your computer.

If you are using the .zip file downloaded from the poLCA website, load R and select `Packages > Install package(s) from local zip files...` from the drop-down menu and navigate to the .zip file. Click `Open` to begin the installation.

Once either installation process is complete, enter

```
> library(poLCA)
```

in R to load the package into memory.

1.2 Data and formula definition

poLCA requires the user to provide a data frame of categorical variables, and a formula definition for the model to be estimated. The data frame may contain missing values (`NA`), but all other entries must be positive integers. Each variable should contain values that increment from 1 to the maximum number of outcome categories for that variable.

Suppose a data frame `dat` contains variables `X1`, `X2`, `Y1`, `Y2`, `Y3`, and `Y4`. To estimate a latent class model for the outcome variables `Y`, define model formula `f`:

```
> f <- cbind(Y1,Y2,Y3,Y4)~1
```

To include covariates, modify the formula using the standard R formula expression:

```
> f <- cbind(Y1,Y2,Y3,Y4)~X1+X2
```

This will estimate the latent class regression model using `X1` and `X2` to predict latent class membership.

1.3 Estimation

To estimate the latent class model with two latent classes (the default), the command is simply:

```
> lc <- poLCA(f,dat)
```

Additional classes can be assumed using the `nclass` argument, as for example:

```
> lc <- poLCA(f,dat,nclass=4)
```

After estimating the model, poLCA will output selected parameters. Other values of interest are saved as a list in `lc`.

1.4 Global versus local maxima

It is always advisable to run poLCA more than once to ensure that the global maximum likelihood of the latent class model has been obtained, rather than only a local maximum. This is due to the algorithm that poLCA uses to estimate the parameters of the latent class model. For more details, see Section 5.5 below.

2 Motivation for poLCA

poLCA is the first R package to enable the user to estimate latent class models for manifest variables with any number of possible outcomes, and it is the only package that estimates latent class regression models with covariates. The two other R commands that currently exist to estimate latent class models—the `lca` command in package e1071, and the `gllm` command in package gllm—can only estimate the basic model for dichotomous outcome variables.

Note that there is occasionally some confusion over the term "latent class regression" (LCR); in practice it can have two meanings. In poLCA, LCR models refer to latent class models in which the probability of latent class membership is predicted by one or more covariates. In other contexts, however, LCR is used to refer to regression models in which the dependent variable is partitioned into 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` command in package fpc will estimate this other type of LCR model, as will the `flexmix` command in package flexmix. Because of these terminology issues, the LCR models estimated using poLCA are sometimes termed "latent class models with covariates" or "concomitant-variable latent class analysis," both of which are accurate descriptions of this model.

3 Latent Class Models

The basic latent class model is a finite mixture model in which the component distributions are assumed to be multi-way cross-classification tables with all variables mutually independent. This assumption is termed "local" or "conditional" independence. This model was originally proposed by Lazarsfeld (1950) under the name "latent structure analysis". Chapter 13 in Agresti (2002) details the connection between latent class models and finite mixture models.

3.1 Terminology

Suppose we observe J unordered polytomous categorical variables (the "manifest" variables), each of which contains K_j possible outcomes, for individuals $i = 1...N$.

The manifest variables may have different numbers of outcomes, hence the indexing by j. Denote as Y_{ijk} the observed values of the J manifest variables such that $Y_{ijk} = 1$ if respondent i gives the kth response to the jth variable, and $Y_{ijk} = 0$ otherwise, where $j = 1 \dots J$ and $k = 1 \dots K_j$.

The latent class model approximates the observed joint distribution of the manifest variables as the weighted sum of a finite number, R, of constituent cross-classification tables. Let π_{jrk} denote the class-conditional probability that an observation in class r produces the kth outcome on the jth variable, where $r = 1 \dots R$. Within each class, for each manifest variable, therefore, $\sum_{k=1}^{K_j} \pi_{jrk} = 1$. Further denote as p_r the R mixing proportions that provide the weights in the weighted sum of the component tables, with $\sum_r p_r = 1$. The number of latent classes R in the model must be specified by the researcher prior to estimating the model.

The probability that an individual i in class r produces a particular set of J outcomes on the manifest variables, assuming local independence, is the product

$$f(Y_i; \pi_r) = \prod_{j=1}^{J} \prod_{k=1}^{K_j} (\pi_{jrk})^{Y_{ijk}}. \qquad (1)$$

The probability density function across all classes is the weighted sum

$$\Pr(Y_i | \pi, p) = \sum_{r=1}^{R} p_r \prod_{j=1}^{J} \prod_{k=1}^{K_j} (\pi_{jrk})^{Y_{ijk}}. \qquad (2)$$

The parameters estimated by the latent class model are p_r and π_{jrk}.

Given estimates $\hat{p}_r$ and $\hat{\pi}_{jrk}$ of p_r and π_{jrk}, respectively, the posterior probability that each individual belongs to each class, conditional on the observed values of the manifest variables, can be calculated using Bayes' formula:

$$\widehat{\Pr}(r | Y_i) = \frac{p_r f(Y_i; \hat{\pi}_r)}{\sum_r p_r f(Y_i; \hat{\pi}_r)}. \qquad (3)$$

Recall that the $\hat{\pi}_{jrk}$ are estimates of outcome probabilities *conditional on* class r.

It is important to remain aware that the number of independent parameters estimated by the latent class model increases rapidly with R, J, and K_j. Given these values, the number of parameters is $R \sum_j (K_j - 1) + (R - 1)$. If this number exceeds either the total number of observations, or one fewer than the total number of cells in the cross-classification table of the manifest variables, then the latent class model will be unidentified.

3.2 Parameter estimation

poLCA estimates the latent class model by maximizing the log-likelihood function

$$\ln L = \sum_{i=1}^{N} \ln \sum_{r=1}^{R} p_r \prod_{j=1}^{J} \prod_{k=1}^{K_j} (\pi_{jrk})^{Y_{ijk}} \qquad (4)$$

with respect to p_r and π_{jrk}, using the expectation-maximization (EM) algorithm (Dempster, Laird, and Rubin 1977). This log-likelihood function is identical in form to the standard finite mixture model log-likelihood. As with any finite mixture model, The EM algorithm is applicable because each individual's class membership is unknown and may be treated as missing data (see McLachlan and Krishnan 1997; McLachlan and Peel 2000).

The EM algorithm proceeds iteratively. Begin with arbitrary initial values of $\hat{p}_r$ and $\hat{\pi}_{jrk}$, and label them $\hat{p}_r^{old}$ and $\hat{\pi}_{jrk}^{old}$. In the expectation step, calculate the "missing" class membership probabilities using Eq. 3, substituting in $\hat{p}_r^{old}$ and $\hat{\pi}_{jrk}^{old}$. In the maximization step, update the parameter estimates by maximizing the log-likelihood function given these posterior $\widehat{\Pr}(r|Y_i)$, with

$$\hat{p}_r^{new} = \frac{1}{N} \sum_{i=1}^{N} \widehat{\Pr}(r|Y_i) \tag{5}$$

as the new prior probabilities and

$$\hat{\pi}_{jr}^{new} = \frac{\sum_{i=1}^{N} Y_{ij} \widehat{\Pr}(r|Y_i)}{\sum_{i=1}^{N} \widehat{\Pr}(r|Y_i)} \tag{6}$$

as the new class-conditional outcome probabilities (see Everitt and Hand 1981; Everitt 1984). In Eq. 6, $\hat{\pi}_{jr}^{new}$ is the vector of length K_j of class-r conditional outcome probabilities for the jth manifest variable; and Y_{ij} is the $N \times K_j$ matrix of observed outcomes Y_{ijk} on that variable. The algorithm repeats these steps, assigning the new to the old, until the overall log-likelihood reaches a maximum and ceases to increment beyond some arbitrarily small value.

poLCA takes advantage of the iterative nature of the EM algorithm to make it possible to estimate the latent class model even when some of the observations on the manifest variables are missing. Although poLCA does offer the option to listwise delete observations with missing values before estimating the model, it is not necessary to do so. Instead, when determining the product in Eq. 1 and the sum in the numerator of Eq. 6, poLCA simply excludes from the calculation any manifest variables with missing observations. The priors are updated in Eq. 3 using as many or as few manifest variables as are observed for each individual.

Depending on the initial values chosen for $\hat{p}_r^{old}$ and $\hat{\pi}_{jrk}^{old}$, and the complexity of the latent class model being estimated, the EM algorithm may only find a local maximum of the log-likelihood function, rather than the preferred global maximum. For this reason, it is always advisable to re-estimate a particular model a couple of times using poLCA, to ensure that the global maximum likelihood solution has been achieved.

3.3 Standard error estimation

poLCA estimates standard errors of the estimated class-conditional response probabilities $\hat{\pi}_{jrk}$ and the mixing parameters $\hat{p}_r$ using the *empirical observed* information matrix (Meilijson 1989), which, following McLachlan and Peel (2000, p. 66), equals

$$\boldsymbol{I_e}(\hat{\Psi}; Y) = \sum_{i=1}^{N} \boldsymbol{s}(Y_i; \hat{\Psi}) \boldsymbol{s}^T(Y_i; \hat{\Psi}), \tag{7}$$

where $\boldsymbol{s}(Y_i; \hat{\Psi})$ is the score function with respect to model parameter Ψ for the ith observation, evaluated at the maximum likelihood estimate $\hat{\Psi}$;

$$\boldsymbol{s}(Y_i; \Psi) = \sum_{r=1}^{R} \theta_{ir} \partial \{\ln p_r + \sum_{j=1}^{J} \sum_{k=1}^{K_j} Y_{ijk} \ln \pi_{jrk}\} / \partial \Psi. \tag{8}$$

The covariance matrix of the parameter estimates is then approximated by the inverse of $\boldsymbol{I_e}(\hat{\Psi}; Y)$.

Because of the sum-to-one constraint on the π_{jrk} across each manifest variable, it is useful to reparameterize the score function in terms of log-ratios $\phi_{jrk} = \ln(\pi_{jrk}/\pi_{jr1})$ for given outcome variable j and class r. Then, for the lth response on the hth item in the qth class,

$$\boldsymbol{s}(Y_i; \phi_{hql}) = \theta_{iq}(Y_{ihl} - \pi_{hql}). \tag{9}$$

Likewise, denoting $\omega_r = \ln(p_r/p_1)$, then for the log-ratio corresponding to the qth mixing parameter,

$$\boldsymbol{s}(Y_i; \omega_q) = \theta_{iq} - p_q. \tag{10}$$

To transform the covariance matrix of these log-ratios back to the original units of π and p, we apply the Delta Method. For the response probabilities, let $g(\phi_{jrk}) = \pi_{jrk} = e^{\phi_{jrk}} / \sum_k e^{\phi_{jrk}}$. Taking as $\mathrm{Var}(\hat{\phi})$ the submatrix of the inverse of $\boldsymbol{I_e}(\hat{\Psi}; Y)$ corresponding to the ϕ parameters, then

$$\mathrm{Var}(g(\hat{\phi})) = g'(\phi) \mathrm{Var}(\hat{\phi}) g'(\phi)^T$$

where $g'(\phi)$ is the Jacobian consisting of elements

$$\frac{\partial g(\phi_{jrk})}{\partial \phi_{hql}} = \begin{cases} 0 & \text{if } q \neq r \\ 0 & \text{if } q = r \text{ but } h \neq j \\ -\pi_{jrk}\pi_{jrl} & \text{if } q = r \text{ and } h = j \text{ but } l \neq k \\ \pi_{jrk}(1 - \pi_{jrk}) & \text{if } q = r \text{ and } h = j \text{ and } l = k. \end{cases}$$

For the mixing parameters, similarly let $h(\omega_r) = p_r = e^{p_r} / \sum_r e^{p_r}$. Taking as $\mathrm{Var}(\hat{\omega})$ the submatrix of the inverse of $\boldsymbol{I_e}(\hat{\Psi}; Y)$ corresponding to the ω parameters, then

$$\mathrm{Var}(h(\hat{\omega})) = h'(\omega) \mathrm{Var}(\hat{\omega}) h'(\omega)^T$$

where $h'(\omega)$ is the Jacobian consisting of elements

$$\frac{\partial h(\omega_r)}{\partial \omega_q} = \begin{cases} -p_r p_q & \text{if } q \neq r \\ p_r(1 - p_r) & \text{if } q = r. \end{cases}$$

Standard errors of each parameter estimate are equal to the square root of the values along the main diagonal of covariance matrices $\mathrm{Var}(\pi)$ and $\mathrm{Var}(p)$.

3.4 Model selection and goodness-of-fit criteria

The choice of number of latent classes is typically either guided by theory, or made with reference to parsimony criteria that are designed to strike a balance between over- and under-fitting the model to the data. Adding an additional class to a latent class model will increase the fit of the model, but at the risk of fitting to noise, and at the expense of estimating a further $1 + \sum_j (K_j - 1)$ model parameters.

Parsimony criteria handle this tradeoff by penalizing the log-likelihood by a function of the number of parameters being estimated. The two most widely used parsimony measures are the Bayesian information criterion (BIC) (Schwartz 1978) and Akaike (1973) information criterion (AIC). Preferred models are those that minimize values of the BIC and/or AIC. Let Λ represent the maximum log-likelihood of the model and Φ represent the total number of estimated parameters. Then,

$$\text{AIC} = -2\Lambda + 2\Phi$$

and

$$\text{BIC} = -2\Lambda + \Phi \ln N.$$

poLCA calculates these parameters automatically when estimating the latent class model. The BIC will usually be more appropriate for basic latent class models because of their relative simplicity (Lin and Dayton 1997; also see Forster 2000).

Calculating χ^2 and likelihood ratio chi-square (G^2) statistics for the observed versus predicted cell counts is another method to determine how well a particular model fits the data (Goodman 1970). Let q_c denote the observed number of cases in each of the $C = \prod K_j$ cells of the cross-classification table of the manifest variables. Let $\hat{Q}$ denote the expected number of cases in each cell under a given model. The cth cell (where $c = 1 \ldots C$) corresponds to one particular sequence of J outcomes on the manifest variables. Taking the $\hat{\pi}_{jrk}$ corresponding *only* to those outcomes,

$$\hat{Q}_c = N \sum_{r=1}^{R} p_r \prod_{j=1}^{J} \hat{\pi}_{jrk}.$$

The two test statistics are then calculated as

$$\chi^2 = \sum_c (q - \hat{Q})^2 / \hat{Q}$$

and

$$G^2 = 2 \sum_c q \ln(q/\hat{Q}).$$

Generally, the goal is to select models that minimize these values without estimating excessive numbers of parameters. Note, however, that the distributional assumptions for these statistics are not met if many cells of the observed cross-classification table contain zero observations. Like the AIC and BIC, these statistics are outputted automatically after calling `poLCA`.

4 Latent Class Regression Models

The latent class regression model generalizes the basic latent class model by permitting the inclusion of covariates (or “concomitant” variables) to predict individuals’ latent class membership (Dayton and Macready 1988; Hagenaars and McCutcheon 2002). This is a so-called “one-step” technique for estimating the effects of covariates, because the coefficients on the covariates are estimated simultaneously as part of the latent class model. An alternate estimation procedure that is sometimes used is called the “three-step” model: estimate the basic latent class model, calculate the predicted posterior class membership probabilities using Eq. 3, and then use these values as the dependent variable(s) in a regression model with the desired covariates. However, as demonstrated by Bolck et al. (2004), the three-step procedure produces biased coefficient estimates. It is preferable to estimate the entire latent class regression model all at once.

Covariates are included in the latent class regression model through their effects on the priors p_r. In the basic latent class model, it is assumed that every individual has the same prior probabilities of latent class membership. The latent class regression model, in contrast, allows individuals’ priors to vary depending upon their observed covariates.

4.1 Terminology

Denote the mixing proportions in the latent class regression model as p_{ri} to reflect the fact that these priors are now free to vary by individual. It is still the case that $\sum_r p_{ri} = 1$ for each individual. To accommodate this constraint, poLCA employs a generalized (multinomial) logit link function for the effects of the covariates on the priors (Agresti 2002).

Let X_i represent the observed covariates for individual i. poLCA arbitrarily selects the first latent class as a “reference” class and assumes that the log-odds of the latent class membership priors with respect to that class are linear functions of the covariates. Let $\boldsymbol{\beta}_r$ denote the vector of coefficients corresponding to the rth latent class. With S covariates, the $\boldsymbol{\beta}_r$ have length $S+1$; this is one coefficient on each of the covariates plus a constant. Because the first class is used as the reference, $\boldsymbol{\beta}_1 = 0$ is fixed by definition. Then,

$$\begin{aligned}
\ln(p_{2i}/p_{1i}) &= X_i\boldsymbol{\beta}_2 \\
\ln(p_{3i}/p_{1i}) &= X_i\boldsymbol{\beta}_3 \\
&\vdots \\
\ln(p_{Ri}/p_{1i}) &= X_i\boldsymbol{\beta}_R
\end{aligned}$$

Following some basic algebra, this produces the general result that

$$p_{ri} = p_r(X_i; \boldsymbol{\beta}) = \frac{e^{X_i \boldsymbol{\beta}_r}}{\sum_{q=1}^{R} e^{X_i \boldsymbol{\beta}_q}}. \tag{11}$$

The parameters estimated by the latent class regression model are the $R-1$ vectors of coefficients $\boldsymbol{\beta}_r$ and, as in the basic latent class model, the class-conditional outcome probabilities π_{jrk}. Given estimates $\hat{\boldsymbol{\beta}}_r$ and $\hat{\pi}_{jrk}$ of these parameters, the posterior class membership probabilities in the latent class regression model are obtained by replacing the p_r in Eq. 3 with the function $p_r(X_i; \boldsymbol{\beta})$ in Eq. 11:

$$\widehat{\Pr}(r|X_i; Y_i) = \frac{p_r(X_i; \hat{\boldsymbol{\beta}}) f(Y_i; \hat{\pi}_r)}{\sum_r p_r(X_i; \hat{\boldsymbol{\beta}}) f(Y_i; \hat{\pi}_r)}. \tag{12}$$

The number of parameters estimated by the latent class regression model is equal to $R \sum_j (K_j - 1) + (S+1)(R-1)$. The same considerations mentioned earlier regarding model identifiability also apply here.

4.2 Parameter estimation

The latent class regression model log-likelihood function is identical to Eq. 4 except that the function $p_r(X_i; \boldsymbol{\beta})$ (Eq. 11) takes the place of p_r:

$$\ln L = \sum_{i=1}^{N} \ln \sum_{r=1}^{R} p_r(X_i; \boldsymbol{\beta}) \prod_{j=1}^{J} \prod_{k=1}^{K_j} (\pi_{jrk})^{Y_{ijk}}. \tag{13}$$

To find the values of $\hat{\boldsymbol{\beta}}_r$ and $\hat{\pi}_{jrk}$ that maximize this function, poLCA uses a modified EM algorithm with a Newton-Raphson step, as set forth by Bandeen-Roche et al. (1997). This estimation process begins with initial values of $\hat{\boldsymbol{\beta}}_r^{old}$ and $\hat{\pi}_{jrk}^{old}$ that are used to calculate posterior probabilities $\widehat{\Pr}(r|X_i; Y_i)$ (Eq. 12). The coefficients on the concomitant variables are updated according to the formula

$$\hat{\boldsymbol{\beta}}_r^{new} = \hat{\boldsymbol{\beta}}_r^{old} + (-\mathbf{D}_{\boldsymbol{\beta}}^2 \ln L)^{-1} \mathbf{D}_{\boldsymbol{\beta}} \ln L \tag{14}$$

where $\mathbf{D}_{\boldsymbol{\beta}}$ is the gradient and $\mathbf{D}_{\boldsymbol{\beta}}^2$ the Hessian matrix with respect to $\boldsymbol{\beta}$. The $\hat{\pi}_{jrk}^{new}$ are updated as

$$\hat{\pi}_{jr}^{new} = \frac{\sum_{i=1}^{N} Y_{ij} \widehat{\Pr}(r|X_i; Y_i)}{\sum_{i=1}^{N} \widehat{\Pr}(r|X_i; Y_i)}. \tag{15}$$

These steps are repeated until convergence, assigning the new parameter estimates to the old in each iteration. The formulas for the gradient and Hessian matrix are provided in Bandeen-Roche et al. (1997).

Because all of the concomitant variables must be observed in order to calculate p_{ri} (Eq. 11), poLCA listwise deletes cases with missing values on the X_i before estimating the latent class regression model. However, missing values on the manifest variables Y_i can be accommodated in the latent class regression model, just as they were in the basic latent class model.

Note that when employing this estimation algorithm, different initial parameter values may lead to different local maxima of the log-likelihood function. To ensure that the global maximum likelihood has been found, the poLCA function call should always be repeated a handful of times.

4.3 Standard error estimation

For latent class models with covariates, standard errors are obtained just as for models without covariates: using the empirical observed information matrix (Eq. 7). First, we generalize the score function (Eq. 8) so that

$$\boldsymbol{s}(X_i, Y_i; \Psi) = \sum_{r=1}^{R} \theta_{ir} \partial \{\ln p_r(X_i; \boldsymbol{\beta}) + \sum_{j=1}^{J} \sum_{k=1}^{K_j} Y_{ijk} \ln \pi_{jrk}\} / \partial \Psi. \tag{16}$$

Since this function is no different than Eq. 8 in terms of the π parameters, the score function $\boldsymbol{s}(X_i, Y_i; \phi_{hql}) = \boldsymbol{s}(Y_i; \phi_{hql})$ (Eq. 9), and the covariance matrix $\mathrm{Var}(\pi)$ may be calculated in precisely the same way as for models without covariates.

Now, however, the priors p_{ri} are free to vary by individual as a function of some set of coefficients $\boldsymbol{\beta}$, as given in Eq. 11. Letting q index classes and s index covariates,

$$s(X_i, Y_i; \beta_{qs}) = X_{is}(\theta_{iq} - p_{iq}). \tag{17}$$

The standard errors of the coefficients β are equal to the square root of the values along the main diagonal of the submatrix of the inverse of the empirical observed information matrix corresponding to the β parameters. (Note that when the model has no covariates, $X_i = 1$ and $p_{iq} = p_q$ (that is, the priors do not vary by individual), so Eq. 17 reduces to Eq. 10 as expected.)

To obtain the covariance matrix of the mixing parameters p_r, which are the average value across all observations of the priors p_{ir}, we apply the Delta Method. Let

$$h(\beta_r) = p_r = \frac{1}{N} \sum_i \left(\frac{e^{X_i \boldsymbol{\beta_r}}}{\sum_{q=1}^{R} e^{X_i \boldsymbol{\beta_q}}} \right).$$

Then

$$\mathrm{Var}(h(\hat{\beta})) = h'(\beta) \mathrm{Var}(\hat{\beta}) h'(\beta)^T$$

where $h'(\beta)$ is a Jacobian with elements

$$\frac{\partial h(\beta_r)}{\partial \beta_{qs}} = \begin{cases} \frac{1}{N} \sum_i X_{is}(-p_{ir} p_{iq}) & \text{if } q \neq r \\ \frac{1}{N} \sum_i X_{is}(p_{ir}(1 - p_{ir})) & \text{if } q = r. \end{cases}$$

5 Using poLCA

The poLCA package makes it possible to estimate a wide range of latent class models in R using a single command line, `poLCA`. Also included in the package is the command `poLCA.simdata`, which enables the user to create simulated data sets that match the data-generating process assumed by either the basic latent class model or the latent class regression model. This functionality is useful for testing the poLCA estimator and for performing Monte Carlo-style analyses of latent class models.

Begin by loading the poLCA package into memory in R by entering

```
> library(poLCA)
```

The internal documentation for the `poLCA` command may be viewed at any time by entering

```
> ? poLCA
```

5.1 Data input and sample data sets

Data are input to the `poLCA` function as a data frame containing all manifest and concomitant variables (if needed). *The manifest variables must be coded as integer values starting at one for the first outcome category, and increasing to the maximum number of outcomes for each variable.* If any of the manifest variables contain zeros, negative values, or decimals, `poLCA` will produce an error message and terminate without estimating the model.

poLCA also comes pre-installed with five sample data sets that are useful for exploring different aspects of latent class and latent class regression models.

`carcinoma`: Dichotomous ratings by seven pathologists of 118 slides for the presence or absence of carcinoma in the uterine cervix. Source: Agresti (2002, p. 542).

`cheating`: Dichotomous responses by 319 undergraduates to questions about cheating behavior. Also each student's GPA, which is useful as a concomitant variable. Source: Dayton (1998, pp. 33 and 85).

`election`: Two sets of six questions with four responses each, asking respondents' opinions of how well various traits describe presidential candidates Al Gore and George W. Bush. Also potential covariates vote choice, age, education, gender, and party ID. Source: 2000 National Election Studies.

`gss82`: Attitudes towards survey taking across two dichotomous and two trichotomous items among 1202 white respondents to the 1982 General Social Survey. Source: McCutcheon (1987, p. 30).

values: Dichotomous measures of 216 survey respondents' tendencies towards "universalistic" or "particularistic" values on four questions. Source: Goodman (1974).

These data sets may be accessed using the `data(`*name*`)` command. Examples of models and analyses using the sample data sets are included in the internal documentation for each.

5.2 poLCA command line options

To specify a latent class model, poLCA uses the standard, symbolic R model formula expression. The response variables are the manifest variables of the model. Because latent class models have multiple manifest variables, these variables must be "bound" as `cbind(Y1,Y2,Y3,...)` in the model formula. For the basic latent class model with no covariates, the formula definition takes the form

```
> f <- cbind(Y1,Y2,Y3)~1
```

The `~1` instructs `poLCA` to estimate the basic latent class model. For the latent class regression model, replace the `~1` with the desired function of covariates, as, for example:

```
> f <- cbind(Y1,Y2,Y3)~X1+X2*X3
```

Further assistance on formula specification in R can be obtained by entering `? formula` at the command prompt.

To estimate the specified latent class model, the default `poLCA` command is:

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

At minimum, it is necessary to enter a formula (as just described) and a data frame (as described in the previous subsection). The remaining options are:

nclass: The number of latent classes to assume in the model; R in the above notation. Setting `nclass=1` results in `poLCA` estimating the loglinear independence model (Goodman 1970). The default is two.

maxiter: The maximum number of iterations through which the estimation algorithm will cycle. If convergence is not achieved before reaching this number of iterations, `poLCA` terminates and reports an error message. The default is 1000, but this will be insufficient for certain models.

graphs: Logical, for whether `poLCA` should graphically display the parameter estimates at each stage of the updating algorithm. The default is `FALSE`, as setting this option to `TRUE` slows down the estimation process.

`tol`: A tolerance value for judging when convergence has been reached. When the one-iteration change in the estimated log-likelihood is less than `tol`, the estimation algorithm stops updating and considers the maximum log-likelihood to have been found. The default is 1×10^{-10} which is a standard value; this option will rarely need to be invoked.

`na.rm`: Logical, for how `poLCA` handles cases with missing values on the manifest variables. If `TRUE`, those cases are removed (listwise deleted) before estimating the model. If `FALSE`, cases with missing values are retained. (As discussed above, cases with missing covariates are always removed.) The default is `TRUE`.

`probs.start`: A list of matrices of class-conditional response probabilities, π_{jrk}, to be used as the starting values for the EM estimation algorithm. Each matrix in the list corresponds to one manifest variable, with one row for each latent class, and one column for each possible outcome. The default is `NULL`, meaning that starting values are generated randomly.

5.3 poLCA output

The `poLCA` function returns an object containing the following elements:

`y`: A data frame of the manifest variables.

`x`: A data frame of the covariates, if specified.

`N`: Number of cases used in the model.

`Nobs`: Number of fully observed cases (less than or equal to `N`).

`probs`: A list of matrices containing the estimated class-conditional outcome probabilities $\hat{\pi}_{jrk}$. Each item in the list represents one manifest variable; columns correspond to possible outcomes on each variable, and rows correspond to the latent classes.

`probs.se`: Standard errors of the estimated class-conditional response probabilities, in the same format as `probs`.

`P`: The respective size of each latent class; equal to the estimated mixing proportions $\hat{p}_r$ in the basic latent class model, or the mean of the priors in the latent class regression model.

`P.se`: The standard errors of `P`.

`posterior`: An $N \times R$ matrix containing each observation's posterior class membership probabilities.

`predclass`: A vector of length N of predicted class memberships, by modal assignment.

`predcell`: A table of observed versus predicted cell counts for cases with no missing values.

`llik`: The maximum value of the estimated model log-likelihood.

`numiter`: The number of iterations required by the estimation algorithm to achieve convergence.

`coeff`: An $(S+1) \times (R-1)$ matrix of estimated multinomial logit coefficients $\hat{\boldsymbol{\beta}}_r$, for the latent class regression model. Rows correspond to concomitant variables X. Columns correspond to the second through Rth latent classes; see Eq. 11.

`coeff.se`: Standard errors of the coefficient estimates, in the same format as `coeff`.

`coeff.V`: Covariance matrix of the coefficient estimates.

`aic`: Akaike Information Criterion.

`bic`: Bayesian Information Criterion.

`Gsq`: Likelihood ratio/deviance statistic.

`Chisq`: Pearson Chi-square goodness of fit statistic.

`time`: Length of time it took to estimate the model.

`npar`: The number of degrees of freedom used by the model (that is, the number of estimated parameters).

`resid.df`: The number of residual degrees of freedom, equal to the lesser of N and $(\prod_j K_j) - 1$, minus `npar`.

`eflag`: Logical, error flag. True if estimation algorithm needed to automatically restart with new initial parameters, otherwise false. A restart is caused in the event of either a non-invertible Hessian matrix, or computational/rounding errors that result in nonsensical parameter estimates. If one of these errors occurs, `poLCA` outputs an error message to alert the user.

`probs.start`: A list of matrices containing the class-conditional response probabilities used as starting values in the EM estimation algorithm. If the algorithm needed to restart (see `eflag`), this contains the starting values used for the final, successful, run of the estimation algorithm.

Selected items from this list are outputted automatically once the specified latent class model has been estimated.

5.4 Reordering the latent classes

Because the latent classes are unordered categories, the numerical order of the estimated latent classes in the model output is arbitrary, and is determined solely by the start 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. This means that repeated runs of `poLCA` will typically produce results containing the same parameter estimates (corresponding to the same maximum log-likelihood), but with reordered latent class labels.

To manually "fix" the order of the estimated latent classes, run `poLCA` once, and extract the outputted list of `probs.start`.

```
> lc <- poLCA(f,dat,nclass=2)
> probs.start <- lc$probs.start
```

Then rearrange the order of the rows (corresponding to the latent classes) in each matrix of that list, to match the desired ordering of the outputted latent classes. For example, suppose you have estimated a two-class model and wish to reverse the class labels in the output.

```
> for (j in 1:length(probs.start)) {
>     probs.start[[j]] <- probs.start[[j]][c(2,1),]
> }
```

Then run `poLCA` once more, this time using the reordered starting values in the function call.

```
> lc <- poLCA(f,dat,nclass=2,probs.start=probs.start)
```

The outputted class labels will now match the desired ordering.

5.5 Recognizing and avoiding local maxima

A well-known drawback of the EM algorithm is that depending upon the initial parameter values chosen in the first iteration, the algorithm may only find a local, rather than the global, maximum of the log-likelihood function (McLachlan and Krishnan 1997). To avoid these local maxima, a user should *always* call `poLCA` at least a couple of times to ensure that the estimated model parameters correspond to the model with the global maximum likelihood.

We demonstrate this using a basic three-class latent class model to analyze the four survey variables in the `gss82` data set included in the poLCA package.

```
> data(gss82)
> f <- cbind(PURPOSE,ACCURACY,UNDERSTA,COOPERAT)~1
```

Maximum	Number of	Respondent Type		
log-likelihood	occurrences	Ideal	Skeptics	Believers
-2754.545	258	0.621	0.172	0.207
-2755.617	14	0.782	0.150	0.067
-2755.739	57	0.796	0.162	0.043
-2762.005	70	0.508	0.392	0.099
-2762.231	101	0.297	0.533	0.170

Table 1: *Results of 500* `poLCA` *function calls for three-class model using* `gss82` *data set. Five local maxima of the log-likelihood function were found. Estimated latent class proportions* $\hat{p}_r$ *are reported for each respondent type at each local maximum.*

We estimate this model 500 times, and after each function call, we record the maximum log-likelihood and the estimated population sizes of the three types of survey respondent. Following McCutcheon (1987), from whom these data were obtained, we label the three types *ideal*, *skeptics*, and *believers*. Among other characteristics, the ideal type is the most likely to have a good understanding of surveys, while the believer type is the least likely.

```
> mlmat <- matrix(NA,nrow=500,ncol=4)
> for (i in 1:500) {
>     gss.lc <- poLCA(f,gss82,nclass=3,maxiter=3000,tol=1e-7)
>     mlmat[i,1] <- gss.lc$llik
>     o <- order(gss.lc$probs$UNDERSTA[,1],decreasing=T)
>     mlmat[i,-1] <- gss.lc$P[o]
> }
```

Results of this simulation are reported in Table 1. Of the five local maxima of the log-likelihood function that were found, the global maximum was obtained in only approximately half of the trials. At the global maximum, the ideal type is estimated to represent 62.1% of the population, with another 17.2% skeptics and 20.7% believers. In contrast, the second-most frequent local maximum was also the lowest of the local maxima, and the parameter estimates corresponding to that “solution” are substantially different: 29.7% ideal types, 53.3% skeptics, and 17.0% believers. This is why it is *essential* to run `poLCA` multiple times until you are *certain* that you have found the parameter estimates that produce the global maximum likelihood solution.

5.6 Creating simulated data sets

The command `polca.simdata` will generate simulated data sets that can be used to examine properties of the latent class and latent class regression model estimators. The properties of the simulated data set are fully customizable, but `polca.simdata` uses the following default arguments in the function call.

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

These input arguments control the following parameters:

`N`: Total number of observations, N.

`probs`: A list of matrices of dimension `nclass` × `nresp`, containing, by row, the class-conditional outcome probabilities π_{jrk} (which must sum to 1) for the manifest variables. Each matrix represents one manifest variable. If `probs` is `NULL` (default) then the outcome probabilities are generated randomly.

`nclass`: The number of latent classes, R. If `probs` is specified, then `nclass` is set equal to the number of rows in each matrix in that list. If `classdist` is specified, then `nclass` is set equal to the length of that vector. Otherwise, the default is two.

`ndv`: The number of manifest variables, J. If `probs` is specified, then `ndv` is set equal to the number of matrices in that list. If `nresp` is specified, then `ndv` is set equal to the length of that vector. Otherwise, the default is four.

`nresp`: The number of possible outcomes for each manifest variable, K_j, entered as a vector of length `ndv`. If `probs` is specified, then `ndv` is set equal to the number of columns in each matrix in that list. If both `probs` and `nresp` are `NULL` (default), then the manifest variables are assigned a random number of outcomes between two and five.

`x`: A matrix of concomitant variables, of dimension `N` × `niv`. If `niv` >0 but `x` is `NULL` (default) then the concomitant variable(s) will be generated randomly. If both `x` and `niv` are entered, then then the number of columns in `x` overrides the value of `niv`.

`niv`: The number of concomitant variables, S. Setting `niv` $=0$ (default) creates a data set assuming no covariates. If `nclass=1` then `niv` is automatically set equal to 0. Unless `x` is specified, all covariates consist of random draws from a standard normal distribution and are mutually independent.

`b`: When using covariates, an `niv`+1 × `nclass`−1 matrix of (multinomial) logit coefficients, $\boldsymbol{\beta}_r$. If `b` is `NULL` (default), then coefficients are generated as random integers between -2 and 2.

`classdist`: A vector of mixing proportions of length `nclass`, corresponding to p_r. `classdist` must sum to 1. Disregarded if `niv`>1 because then `classdist` is, in part, a function of the concomitant variables. If `classdist` is `NULL` (default), then the p_r are generated randomly.

`missval`: Logical. If `TRUE` then a fraction `pctmiss` of the observations on the manifest variables are randomly dropped as missing values. Default is `FALSE`.

`pctmiss`: The percentage of values to be dropped as missing, if `missval=TRUE`. If `pctmiss` is `NULL` (default), then a value between 5% and 40% is chosen randomly.

Note that in many instances, specifying values for certain arguments will override other specified arguments. Be sure when calling `polca.simdata` that all arguments are in logical agreement, or else the function may produce unexpected results.

Specifying the list of matrices `probs` can be tricky; we recommend a command structure such as this for, for example, five manifest variables, three latent classes, and $K_j = (3, 2, 3, 4, 3)$.

```
> 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=T),
  matrix(c(0.2,0.8,         0.7,0.3,         0.3,0.7        ),ncol=2,byrow=T),
  matrix(c(0.3,0.6,0.1,     0.1,0.3,0.6,     0.3,0.6,0.1    ),ncol=3,byrow=T),
  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=T),
  matrix(c(0.1,0.1,0.8,     0.1,0.8,0.1,     0.8,0.1,0.1    ),ncol=3,byrow=T))
```

The object returned by `polca.simdata` is a list containing both the simulated data set *and* all of the parameters used to generate that data set. The elements listed here have the same characteristics and meanings as just described.

`dat`: A data frame containing the simulated variables X and Y. Variable names for manifest variables are `Y1`, `Y2`, ..., `Y`J. Variable names for concomitant variables are `X1`, `X2`, ..., `X`S.

`probs`: A list of matrices of dimension `nclass` × `nresp` containing the class-conditional outcome probabilities.

`nresp`: A vector containing the number of possible outcomes for each manifest variable.

`b`: A matrix containing the coefficients on the covariates, if used.

`classdist`: The mixing proportions corresponding to each latent class.

`pctmiss`: The percent of observations missing.

`trueclass`: A vector of length `N` containing the "true" class membership for each individual.

Examples of possible uses of `polca.simdata` are included in the poLCA internal documentation and may be accessed by entering `? poLCA.simdata` in R.

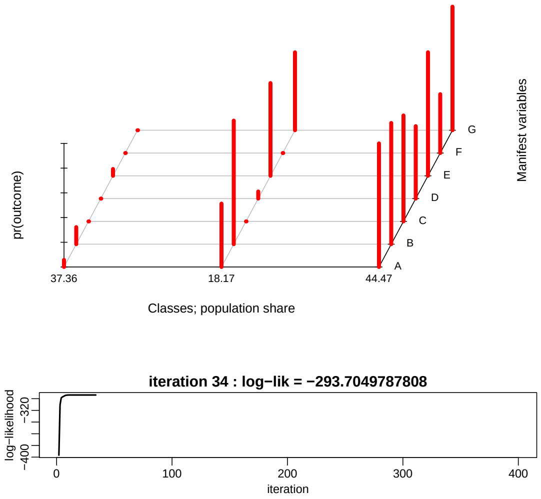

Figure 1: *Estimation of the three-class basic latent class model using the* `carcinoma` *data; obtained by setting* `graphs=TRUE` *in the* `poLCA` *function call. Each group of red bars represents the conditional probabilities, by latent class, of being rated positively by each of the seven pathologists (labeled A through G). Taller bars correspond to conditional probabilities closer to 1 of a positive rating.*

6 Two Examples

To illustrate the usage of the poLCA package, we present two examples: a basic latent class model and a latent class regression model, using sample data sets included in the package.

6.1 Basic latent class modeling with the `carcinoma` data

The `carcinoma` data from Agresti (2002, p. 542) contain seven dichotomous manifest variables that represent the ratings by seven pathologists of 118 slides on the presence or absence of carcinoma. The purpose of studying these data is to model "interobserver agreement" by examining how subjects might be divided into groups depending on the consistency of their diagnoses.

It is straightforward to replicate Agresti's published results (p. 543) using the series of commands:

```
> data(carcinoma)
> f <- cbind(A,B,C,D,E,F,G)~1
> lc2 <- poLCA(f,carcinoma,nclass=2)
> lc3 <- poLCA(f,carcinoma,nclass=3)
> lc4 <- poLCA(f,carcinoma,nclass=4,maxiter=5000)
```

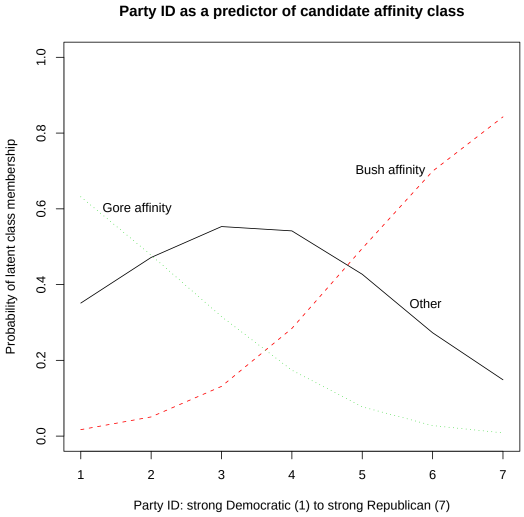

Figure 2: *Predicted prior probabilities of latent class membership at varying levels of partisan self-identification. Results are from a three-class latent class regression model.*

Note that the four-class model will typically require a larger number of iterations to achieve convergence.

Figure 1 shows a screen capture of the estimation of model `lc3` with the `graphs` option set to `TRUE` and `maxiter=400`. In this case, the model has converged after 34 iterations. As Agresti describes, the three estimated latent classes clearly correspond to a pair of classes that are consistently rated negative (37%) or positive (44%), plus a third "problematic" class representing 18% of the population. In that class, pathologists B, E, and G tend to diagnose positive; C, D, and F tend to diagnose negative; and A is about 50/50.

6.2 Latent class regression modeling with the `election` data

In the `election` data set, respondents to the 2000 American National Election Study public opinion poll were asked to evaluate how well a series of traits—moral, caring, knowledgable, good leader, dishonest, and intelligent—described presidential candidates Al Gore and George W. Bush. Each question had four possible choices: (1) extremely well; (2) quite well; (3) not too well; and (4) not well at all.

A reasonable theoretical approach might suppose that there are three latent classes of survey respondents: Gore supporters, Bush supporters, and those who are more or less neutral. Gore supporters will tend to respond favorably towards Gore and unfavorably towards Bush, with the reverse being the case for Bush supporters. Those in the neutral group will not have strong opinions about either candidate. We might further expect that falling into one of these three groups is a function of each individual's party identification, with committed Democrats more likely to favor Gore, committed Republicans more likely to favor Bush, and less intense partisans tending to be indifferent. We can investigate this hypothesis using a latent class regression model.

Begin by loading the election data into memory, and specifying a model with 12 manifest variables and PARTY as the lone concomitant variable. Next, estimate the latent class regression model and assign those results to object nes2a.

```
> data(election)
> f2a <- cbind(MORALG,CARESG,KNOWG,LEADG,DISHONG,INTELG,
               MORALB,CARESB,KNOWB,LEADB,DISHONB,INTELB)~PARTY
> nes2a <- poLCA(f2a,election,nclass=3)
```

The model finds that the three groups indeed separate as expected, with 27% in the favor-Gore group, 34% in the favor-Bush group, and 39% in the neutral group.

To interpret the generalized logit coefficients $\hat{\boldsymbol{\beta}}_r$ estimated by the model, we plot predicted values of p_{ri}, the prior probability of class membership, at varying levels of party ID. The PARTY variable is coded across seven categories, from strong Democrat at 1 to strong Republican at 7. People who primarily consider themselves Independents are at 3-4-5 on the scale. The R commands to do this are as follows, producing the graph in Figure 2.

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

Strong Democrats have over a 60% prior probability of belonging to the Gore affinity group, while strong Republicans have over an 80% prior probability of belonging to the Bush affinity group. The prior probability of belonging to the indifferent category, labeled "Other", is greatest for self-identified Independents (4) and Independents who lean Democratic (3).

7 License, Contact, Versioning, Development

poLCA is provided free of charge, subject to version 2 of the GPL or any later version. Users of poLCA are requested to cite the software package as:

Linzer, Drew A. and Jeffrey Lewis. 2007. “poLCA: Polytomous Variable Latent Class Analysis.” R package version 1.0. `http://dlinzer.bol.ucla.edu/poLCA`.

Please direct all inquiries, comments, and reports of bugs to `dlinzer@ucla.edu`.

7.1 Version history

1.0: Provides standard errors for all model parameters, and covariance matrix for regression model coefficients. Also allows users to specify the starting parameters for the estimation algorithm, to aid in convergence and increase control over model output. (April 4, 2007)

0.9: First public release. (June 1, 2006)

7.2 Planned developments

- An internal looping mechanism to re-estimate the latent class model a user-specified number of multiple times, to help ensure that the global maximum of the log-likelihood function has been found.
- Flexibility to relax the assumption of local independence among user-specified manifest variables in the component cross-classification tables.
- Accommodation of user-specified constraints on selected parameters π_{jrk}, either for theoretical reasons, or to reduce the number of parameters needing to be estimated in the latent class model (Goodman 1974).
- More aggressive error checking on input data, to ensure that manifest variables are entered properly as integers from one to the maximum number of outcomes for each variable, with no zeros or negative numbers.

References

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

Akaike, H. 1973. “Information theory and an extension of the maximum likelihood principle.” In B.N. Petrov and F. Csake, eds. *Second International Symposium on Information Theory*, 267-281. Budapest, Hungary: Akademiai Kiado.

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.

Bolck, Annabel, Marcel Croon and Jacques Hagenaars. 2004. “Estimating Latent Structure Models with Categorical Variables: One-Step Versus Three-Step Estimators.” *Political Analysis.* 12(1): 3-27.

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

Dayton, C. Mitchell and George B. Macready. 1988. "Concomitant-Variable Latent-Class Models." *Journal of the American Statistical Association.* 83(401): 173-178.

Dempster, A.P., N.M. Laird, and D.B. Rubin. 1977. "Maximum likelihood from incomplete data via the EM algorithm (with discussion)." *Journal of the Royal Statistical Society B.* 39: 1-38.

Everitt, B.S. 1984. *An Introduction to Latent Variable Models.* London: Chapman and Hall.

Everitt, B.S. and D.J. Hand. 1981. *Finite Mixture Distributions.* London: Chapman and Hall.

Forster, Malcolm R. 2000. "Key Concepts in Model Selection: Performance and Generalizability." *Journal of Mathematical Psychology.* 44:205-231.

Goodman, Leo. 1970. "The Multivariate Analysis of Qualitative Data: Interactions among Multiple Classifications." *Journal of the American Statistical Association.* 65: 226-256.

Goodman, Leo. 1974. "Exploratory latent structure analysis using both identifiable and unidentifiable models." *Biometrika.* 61: 315-231.

Hagenaars, Jacques A. and Allan L. McCutcheon, eds. 2002. *Applied Latent Class Analysis.* Cambridge: Cambridge University Press.

Lazarsfeld, Paul F. 1950. "The Logical and Mathematical Foundations of Latent Structure Analysis." In Samuel A. Stouffer, ed. *Measurement and Prediction*, 362-412. New York: John Wiley & Sons.

Lin, Ting Hsiang and C. Mitchell Dayton. 1997. "Model Selection Information Criteria for Non-Nested Latent Class Models." *Journal of Educational and Behavioral Statistics.* 22(3): 249-264.

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

McLachlan, Geoffrey J. and Thriyambakam Krishnan. 1997. *The EM Algorithm and Extensions.* New York: John Wiley & Sons, Inc.

McLachlan, Geoffrey and David Peel. 2000. *Finite Mixture Models.* New York: John Wiley & Sons, Inc.

Meilijson, Isaac. 1989. "A Fast Improvement to the EM Algorithm on its Own Terms." *Journal of the Royal Statistical Society. Series B.* 51(1): 127-138.

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].

Schwartz, G. 1978. "Estimating the dimension of a model." *The Annals of Statistics.* 6: 461-464.