poLCA: An **R** Package for Polytomous Variable Latent Class Analysis

Drew A. Linzer
Emory University

Jeffrey B. Lewis
University of California,
Los Angeles

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.

Keywords: latent class analysis, latent class regression, polytomous, categorical, concomitant.

1. Introduction

Latent class analysis is a statistical technique for the analysis of multivariate categorical data. When observed data take the form of a series of categorical responses—as, for example, in public opinion surveys, individual-level voting data, studies of inter-rater reliability, or consumer behavior and decision-making—it is often of interest to investigate sources of confounding between the observed variables, identify and characterize clusters of similar cases, and approximate the distribution of observations across the many variables of interest. Latent class models are a useful tool for accomplishing these goals.

The latent class model seeks to stratify the cross-classification table of observed (or, "manifest") variables by an unobserved ("latent") unordered categorical variable that eliminates all confounding between the manifest variables. Conditional upon values of this latent variable, responses to all of the manifest variables are assumed to be statistically independent;

an assumption typically referred to as "conditional" or "local" independence. The model, in effect, probabilistically groups each observation into a "latent class," which in turn produces expectations about how that observation will respond on each manifest variable. Although the model does not automatically determine the number of latent classes in a given data set, it does offer a variety of parsimony and goodness of fit statistics that the researcher may use in order to make a theoretically and empirically sound assessment.

Because the unobserved latent variable is nominal (membership of a class), the latent class model is actually a type of finite mixture model. The component distributions in the mixture are cross-classification tables of equal dimension to the observed table of manifest variables, and, following the assumption of conditional independence, the frequency in each cell of each component table is simply the product of the respective class-conditional marginal frequencies (the parameters estimated by the latent class model are the proportion of observations in each latent class, and the probabilities of observing each response to each manifest variable, conditional on latent class). A weighted sum of these component tables forms an approximation (or, density estimate) of the distribution of cases across the cells of the observed table. Observations with similar sets of responses on the manifest variables will tend to cluster within the same latent classes. The model may also be fit to manifest variables that are ordinal, but they will be treated as nominal. In practice, this does not usually restrict analyses in any meaningful way.

An extension of this basic model permits the inclusion of covariates to predict latent class membership. Whereas in the basic model, every observation has the same probability of belonging to each latent class prior to observing the responses to the manifest variables, in the more general latent class "regression" model, these prior probabilities vary by individual as a function of some set of independent (or, "concomitant") variables.

Examples of latent class models in political science include McCutcheon (1985), Feick (1989), Breen (2000), Hill and Kriesi (2001a,b), Blaydes and Linzer (2008), and Linzer (2011). The latent class model is similar to the latent trait model widely used for the estimation of voter and legislator "ideal points" (e.g., Clinton, Jackman, and Rivers 2004) in that both models assume the presence of an underlying, unobserved latent variable to explain patterns among observed variables. Yet whereas the latent trait model assumes that the latent variable is continuous, the latent class model assumes that the latent variable is categorical. Moreover, unlike the ideal point model, the latent class model requires no assumptions about respondent utility functions, utility maximization, or rationality.

poLCA is the most complete and most user-friendly package for the estimation of latent class models and latent class regression models in R (Linzer and Lewis 2011; R Development Core Team 2010). The package is available from both the Comprehensive R Archive Network at `http://CRAN.R-project.org/package=poLCA` and the **poLCA** project Web site at `http://userwww.service.emory.edu/~dlinzer/poLCA`. Other R functions for the estimation of latent class models include `lca` in package **e1071**, `gllm` in package **gllm**, and `randomLCA` in package **randomLCA**, but these can only estimate the basic model for dichotomous outcome variables (Duffy 2010; Beath 2011; Dimitriadou, Hornik, Leisch, Meyer, and Weingessel 2011). Package **flexmix** has the capacity to estimate latent class regression models, but not without considerable extra programming effort on the part of the user (Grün and Leisch 2008).

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` function in package **fpc** (Hennig 2010) will estimate this other type of LCR model, as will the `flexmix` function in package **flexmix** (Leisch 2004; Grün and Leisch 2008). 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.

2. 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 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.

2.1. Terminology and model definition

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

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. R is fixed prior to estimation on the basis of either theoretical reasons or model fit; this issue is addressed in greater detail in Section 2.4 below. Let π_{jrk} denote the class-conditional probability that an observation in class $r = 1, \ldots, R$ produces the kth outcome on the jth variable. 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 values of p_r are also referred to as the "prior" probabilities of latent class membership, as they represent the unconditional probability that an individual will belong to each class before taking into account the responses Y_{ijk} provided on the manifest variables.

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

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

The probability density function across all classes is the weighted sum

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

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:

$$\widehat{\mathsf{P}}(r_i|Y_i) = \frac{\hat{p}_r f(Y_i; \hat{\pi}_r)}{\sum_{q=1}^{R} \hat{p}_q f(Y_i; \hat{\pi}_q)}. \tag{3}$$

where $r_i \in \{1, \dots, R\}$. 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.

2.2. Parameter estimation

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

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

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 (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 Equation 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 $\widehat{\mathsf{P}}(r_i|Y_i)$, with

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

as the new prior probabilities and

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

as the new class-conditional outcome probabilities; see Everitt and Hand (1981), and Everitt (1984). In Equation 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 Equation 1 and the sum in the numerator of Equation 6, **poLCA** simply excludes from the calculation any manifest variables with missing observations. The priors are updated in Equation 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 desired global maximum. For this reason, it is always advisable to re-estimate a particular model a couple of times when using **poLCA**, in an attempt to find the global maximizer to be taken as the maximum likelihood solution.

2.3. 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, 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 the vector of parameters Ψ for the ith observation, evaluated at the maximum likelihood estimate $\hat{\Psi}$;

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

where $\theta_{ir} = \widehat{\mathsf{P}}(r_i | Y_i)$ is the posterior probability that observation i belongs to class r (Equation 3). 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 $g(\phi_{jrk}) = \pi_{jrk} = e^{\phi_{jrk}} / \sum_l e^{\phi_{jrl}}$. Taking as $\mathsf{VAR}(\hat{\phi})$ the submatrix of the inverse of $\boldsymbol{I_e}(\hat{\Psi}; Y)$ corresponding to the ϕ parameters, hen

$$\mathsf{VAR}(g(\hat{\phi})) = g'(\phi) \mathsf{VAR}(\hat{\phi}) g'(\phi)^T$$

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 $h(\omega_r) = p_r = e^{p_r}/\sum_q e^{p_q}$. Taking as $\mathsf{VAR}(\hat{\omega})$ the submatrix of the inverse of $\boldsymbol{I_e}(\hat{\Psi}; Y)$ corresponding to the ω parameters, then

$$\mathsf{VAR}(h(\hat{\omega})) = h'(\omega)\mathsf{VAR}(\hat{\omega})h'(\omega)^T$$

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 $\mathsf{VAR}(\pi)$ and $\mathsf{VAR}(p)$.

2.4. Model selection and goodness of fit criteria

One of the benefits of latent class analysis, in contrast to other statistical techniques for clustered data, is the variety of tools available for assessing model fit and determining an appropriate number of latent classes R for a given data set. In some applications, the number of latent classes will be selected for primarily theoretical reasons. In other cases, however, the analysis may be of a more exploratory nature, with the objective being to locate the best fitting or most parsimonious model. The researcher may then begin by fitting a complete "independence" model with $R = 1$, and then iteratively increasing the number of latent classes by one until a suitable fit has been achieved.

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 seek to strike a balance between over- and under-fitting the model to the data 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, or BIC (Schwartz 1978) and Akaike information criterion, or AIC (Akaike 1973). 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; Forster 2000).

Calculating Pearson's χ^2 goodness of fit and likelihood ratio chi-square (G^2) statistics for the observed versus predicted cell counts is another method to help determine how well a particular model fits the data (Goodman 1970). Let q_c denote the observed number of cases in the cth cell of the cross-classification table of the manifest variables, for cells $c = 1 \ldots C$, where $C = \prod K_j$. The expected percentage of the population in each cell of the fitted J-dimensional table is calculated by inserting estimates $\hat{p}_r$ and $\hat{\pi}_{jrk}$ into Equation 2. Denote as y_c the sequence of J outcomes corresponding to the cth cell in the fitted contingency table, such that $y_{cjk} = 1$ if cell c contains the kth response on the jth variable, and $y_{cjk} = 0$ otherwise. Then, the estimated probability mass function produced by the latent class model is

$$\tilde{P}(y_c) = \sum_{r=1}^{R} \hat{p}_r \prod_{j=1}^{J} \prod_{k=1}^{K_j} (\hat{\pi}_{jrk})^{y_{cjk}}. \tag{11}$$

The expected number of cases in each cell under a given model is $\tilde{Q}_c = N\tilde{P}(y_c)$. The two test statistics are

$$\chi^2 = \sum_{c=1}^{C} (q_c - \tilde{Q}_c)^2 / \tilde{Q}_c$$

and

$$G^2 = 2 \sum_{c=1}^{C} q_c \log(q_c / \tilde{Q}_c).$$

Like the AIC and BIC, these statistics are outputted automatically after calling `poLCA`.

Generally, the goal is to select models that minimize χ^2 or G^2 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 very few observations. Common practice holds that no fewer than 10% to 20% of the cells should contain fewer than five observations if either chi-square test is to be used.

3. Latent class regression models

The latent class regression model generalizes the basic latent class model by permitting the inclusion of covariates 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 alternative estimation procedure that is sometimes used is called the "three-step" approach: estimate the basic latent class model, calculate the predicted posterior class membership probabilities using Equation 3, and then use these values as the dependent variable(s) in a regression model with the desired covariates. However, as demonstrated by Bolck, Croon, and Hagenaars (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.

3.1. Terminology and model definition

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,

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

Following some simple algebra, this produces the general result that

$$p_{ri} = p_r(X_i; \boldsymbol{\beta}) = \frac{e^{X_i \boldsymbol{\beta}_r}}{\sum_{q=1}^{R} e^{X_i \boldsymbol{\beta}_q}}. \tag{12}$$

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 Equation 3 with the function $p_r(X_i; \boldsymbol{\beta})$ in Equation 12:

$$\widehat{\mathsf{P}}(r_i | X_i; Y_i) = \frac{p_r(X_i; \hat{\boldsymbol{\beta}}) f(Y_i; \hat{\pi}_r)}{\sum_{q=1}^{R} p_q(X_i; \hat{\boldsymbol{\beta}}) f(Y_i; \hat{\pi}_q)}. \tag{13}$$

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.

3.2. Parameter estimation

The latent class regression model log-likelihood function is identical to Equation 4 except that the function $p_r(X_i; \boldsymbol{\beta})$ (Equation 12) takes the place of p_r:

$$\ln L = \sum_{i=1}^{N} \ln \sum_{r=1}^{R} p_r(X_i; \boldsymbol{\beta}) \prod_{j=1}^{J} \prod_{k=1}^{K_j} (\pi_{jrk})^{Y_{ijk}}. \tag{14}$$

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, Miglioretti, Zeger,

and Rathouz (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 $\widehat{\mathsf{P}}(r_i|X_i; Y_i)$ (Equation 13). The coefficients on the concomitant variables are updated according to the formula

$$\hat{\boldsymbol{\beta}}_r^{new} = \hat{\boldsymbol{\beta}}_r^{old} - \mathbf{H}_{\boldsymbol{\beta}}^{-1}\boldsymbol{\nabla}_{\boldsymbol{\beta}} \tag{15}$$

where $\boldsymbol{\nabla}_{\boldsymbol{\beta}}$ is the gradient and $\mathbf{H}_{\boldsymbol{\beta}}$ the Hessian matrix of the log-likelihood function with respect to $\boldsymbol{\beta}$. The $\hat{\pi}_{jrk}^{new}$ are updated as

$$\hat{\pi}_{jr}^{new} = \frac{\sum_{i=1}^{N} Y_{ij}\widehat{\mathsf{P}}(r_i|X_i; Y_i)}{\sum_{i=1}^{N} \widehat{\mathsf{P}}(r_i|X_i; Y_i)}. \tag{16}$$

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} (Equation 12), **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 be more certain that the global maximum likelihood solution has been found, the `poLCA` function call should always be repeated a handful of times.

3.3. 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 (Equation 7). First, we generalize the score function (Equation 8) so that

$$\boldsymbol{s}(X_i, Y_i; \Psi) = \sum_{r=1}^{R} \theta_{ir}\partial\{\ln p_r(X_i; \boldsymbol{\beta}) + \sum_{j=1}^{J}\sum_{k=1}^{K_j} Y_{ijk}\ln \pi_{jrk}\}/\partial\Psi. \tag{17}$$

As before, θ_{ir} denote posterior probabilities. Since this function is no different than Equation 8 in terms of the π parameters, the score function $\boldsymbol{s}(X_i, Y_i; \phi_{hql}) = \boldsymbol{s}(Y_i; \phi_{hql})$ (Equation 9), and the covariance matrix $\mathsf{VAR}(\pi)$ 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 Equation 12. Letting q index classes and s index covariates,

$$s(X_i, Y_i; \beta_{qs}) = X_{is}(\theta_{iq} - p_{iq}). \tag{18}$$

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 Equation 18 reduces to Equation 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

$$\mathsf{VAR}(h(\hat{\beta})) = h'(\beta)\mathsf{VAR}(\hat{\beta})h'(\beta)^T$$

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

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

When estimating latent class models with covariates, poLCA will compute and automatically display t and p-values for the coefficient estimates, corresponding to a null hypothesis of $\beta_{qs} = 0$.

4. 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.

4.1. 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. The input data frame may contain missing values.

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, 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, 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: The National Election Studies (2000).

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, 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 command `data("carcinoma")`, for example. Examples of models and analyses using the sample data sets are included in the internal documentation for each.

4.2. 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

```
R> 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:

```
R> 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:

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

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 the completion of the estimation algorithm. The default is `FALSE`.

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. Note that if `nrep` > 1, then any user-specified `probs.start` values are only used in the first of the `nrep` attempts.

nrep: Number of times to estimate the model, using different values of `probs.start`. The default is one. Setting `nrep` > 1 automates the search for the global—rather than just a local—maximum of the log-likelihood function. `poLCA` returns only the parameter estimates corresponding to the model producing the greatest log-likelihood.

verbose: Logical, indicating whether `poLCA` should output to the screen the results of the model. If `FALSE`, no output is produced. The default is `TRUE`.

calc.se: Logical, indicating whether `poLCA` should calculate the standard errors of the estimated class-conditional response probabilities and mixing proportions. The default is `TRUE`; can only be set to `FALSE` if estimating a basic model with no concomitant variables specified in `formula`.

4.3. 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. Also see the function **poLCA.posterior**.

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. Also see functions **poLCA.table** and **poLCA.predcell**.

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

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

maxiter: The maximum number of iterations through which the estimation algorithm was set to run.

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 Equation 12.

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**.

attempts: A vector containing the maximum log-likelihood values found in each of the **nrep** attempts to fit the model.

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 computational/rounding errors that result in nonsensical parameter estimates. If an error 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.

`probs.start.ok:` Logical. `FALSE` if `probs.start` was incorrectly specified by the user, otherwise `TRUE`.

If `verbose=TRUE`, selected items from this list are displayed automatically once the latent class model has been estimated.

4.4. Predicted cell frequencies from the latent class model

The `poLCA` object contains an element `predcell` which enables quick comparisons of the observed cell counts to the cell counts predicted by the latent class model—but only for cells that were observed to contain at least one observation. To generate predicted cell counts for *any* combination of the manifest variables, including cells with zero observations, apply the `poLCA.table` function to the fitted latent class model stored in the `poLCA` object. This function post-processes the latent class model estimates to produce frequency distributions and two-way tables of predicted cell counts, holding the values of the other manifest variables fixed at a user-specified set of values.

As an example, consider a basic two-class latent class model fitted to the four survey variables in the `gss82` data set included in the **poLCA** package.

```
R> data("gss82")
R> f <- cbind(PURPOSE, ACCURACY, UNDERSTA, COOPERAT) ~ 1
R> gss.lc2 <- poLCA(f, gss82, nclass = 2)
```

Entering `gss.lc2$predcell` shows that of the 36 possible four-response sequences of responses ($3 \times 2 \times 2 \times 3$), only 33 are actually observed. One unobserved sequence is the combination `PURPOSE`=3, `ACCURACY`=1, `UNDERSTA`=2, `COOPERAT`=3. We produce the predicted frequency table for `COOPERAT` conditional on the specified values of the other three variables using the command

```
R> poLCA.table(formula = COOPERAT ~ 1,
+    condition = list(PURPOSE = 3, ACCURACY = 1, UNDERSTA = 2),
+    lc = gss.lc2)
```

Fitted values of 4.94 for `COOPERAT`=1 and 0.76 for `COOPERAT`=2 also appeared in the outputted `gss.lc2$predcell`; we may now also see that the fitted value for `COOPERAT`=3 is 0.16, close to the observed value of zero.

To make a two-way table, modify the specification of the `formula` argument to contain both a row (`COOPERAT`) and a column (`UNDERSTA`) variable, again using the `condition` argument to hold fixed the values of the other two variables:

```
R> poLCA.table(formula = COOPERAT ~ UNDERSTA,
+    condition = list(PURPOSE = 3, ACCURACY = 1),
+    lc = gss.lc2)
```

The second column of this table, corresponding to UNDERSTA=2, is identical to the conditional frequency just shown.

If the quantity of interest is not the predicted cell *counts*, but rather the cell *percentages* $\tilde{P}(y_c)$ estimated by the model for particular combinations of the manifest variables, apply the poLCA.predcell function. The fitted latent class model is a density estimate of the joint distribution of the manifest variables in the population, which may be represented as a large multi-way contingency table (see, e.g., Linzer 2011). The poLCA.predcell function calculates the value of the estimated probability mass function for specified cells in that table. For example, the latent class model density estimate of the percentage of people in the underlying population replying 1 to all four questions is 34%:

```
R> poLCA.predcell(lc = gss.lc2, y = c(1, 1, 1, 1))
```

Multiplying this percentage by the total number of observations in the data set, 1202, produces an expected cell frequency of 408.1, as may also be seen in the first row of gss.lc2$predcell.

4.5. Entropy of a fitted latent class model

Entropy is a measure of dispersion (or concentration) in a probability mass function. For multivariate categorical data, it is calculated $H = -\sum_c p_c \ln(p_c)$ where p_c is the share of the probability in the cth cell of the cross-classification table. A fitted latent class model produces a smoothed density estimate of the underlying distribution of cell percentages in the multi-way table of the manifest variables. The poLCA.entropy function calculates the entropy of that estimated probability mass function, setting $p_c = \tilde{P}(y_c)$ in the above notation.

4.6. 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 change the order of the latent classes, it is convenient to use the included function poLCA.reorder. Suppose you have estimated a three-class model and wish to reverse the second and third class labels in the output. After an initial call to poLCA, extract the outputted list of probs.start.

```
R> lc <- poLCA(f, dat, nclass = 3)
R> probs.start <- lc$probs.start
```

The poLCA.reorder function takes as its first argument the list of starting values probs.start, and as its second argument a vector describing the desired reordering of the latent classes. In this example, the vector c(1, 3, 2) instructs poLCA.reorder to keep the first class in its current position, but move the third class to the second, and the second class to the third.

```
R> new.probs.start <- poLCA.reorder(probs.start, c(1, 3, 2))
```

Maximum log-likelihood	Number of occurrences	Respondent type		
		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.

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

```
R> lc <- poLCA(f, dat, nclass = 3, probs.start = new.probs.start)
```

The outputted class labels will now match the desired ordering.

4.7. 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* either 1) call `poLCA` at least a couple of times; or 2) utilize the `nrep` argument to attempt to locate the parameter values that globally maximize the log-likelihood function.

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.

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

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.

```
R> mlmat <- NULL
R> for (i in 1:500) {
+    gss.lc <- poLCA(f, gss82, nclass = 3, maxiter = 3000, verbose = FALSE)
+    o <- order(gss.lc$probs$UNDERSTA[, 1], decreasing = TRUE)
+    mlmat <- rbind(mlmat, c(gss.lc$llik, gss.lc$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 can be reasonably certain that you have found the parameter estimates that produce the global maximum likelihood solution.

To automate this search using the nrep argument, specify the model as

```
R> gss.lc <- poLCA(f, gss82, nclass = 3, maxiter = 3000, nrep = 10)
```

The latent class model will be estimated ten times using different initial parameter values, and will assign to gss.lc the results corresponding to the model with the greatest value of the log-likelihood function. Sample output will appear as follows.

```
Model 1: llik = -2762.231 ... best llik = -2762.231
Model 2: llik = -2755.739 ... best llik = -2755.739
Model 3: llik = -2754.545 ... best llik = -2754.545
Model 4: llik = -2754.545 ... best llik = -2754.545
Model 5: llik = -2754.545 ... best llik = -2754.545
Model 6: llik = -2762.005 ... best llik = -2754.545
Model 7: llik = -2755.739 ... best llik = -2754.545
Model 8: llik = -2754.545 ... best llik = -2754.545
Model 9: llik = -2754.545 ... best llik = -2754.545
Model 10: llik = -2754.545 ... best llik = -2754.545
```

In this example, the global maximum log-likelihood, -2754.545, is found in the third attempt at fitting the model.

4.8. 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.

```
R> poLCA.simdata(N = 5000, probs = NULL, nclass = 2, ndv = 4, nresp = NULL,
+    x = NULL, niv = 0, b = NULL, P = 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 P 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, β_r. If `b` is NULL (default), then coefficients are generated as random integers between -2 and 2.

P: A vector of mixing proportions of length `nclass`, corresponding to p_r. P must sum to 1. Disregarded if `niv`> 1 because then P is, in part, a function of the concomitant variables. If P 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 $K_j = (3, 2, 3, 4, 3)$.

```
R> 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))
```

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, …, YJ. Variable names for concomitant variables are X1, X2, …, XS.

probs: A list of matrices of dimension `nclass` $\times$ `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.

P: 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. One example demonstrates that even when the "true" data generating process involves a series of covariates—so that each observation has a different prior probability of belonging to each class—the posterior probabilities of latent class membership can still be recovered with high accuracy using a basic model specified without covariates. A second example confirms that in data sets with missing values, the `poLCA` function produces consistent estimates of the class-conditional response probabilities π_{jrk} regardless of whether the researcher elects to include or listwise delete the observations with missing values.

5. 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.

5.1. Basic latent class modeling with the `carcinoma` data

The `carcinoma` data from Agresti (2002, 542) consist of seven dichotomous 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 upon the consistency of their diagnoses.

It is straightforward to replicate Agresti's published results (Agresti 2002, 543) using the series of commands:

```
R> data("carcinoma")
R> f <- cbind(A, B, C, D, E, F, G) ~ 1
R> lc2 <- poLCA(f, carcinoma, nclass = 2)
R> lc3 <- poLCA(f, carcinoma, nclass = 3, graphs = TRUE)
R> lc4 <- poLCA(f, carcinoma, nclass = 4, maxiter = 5000)
```
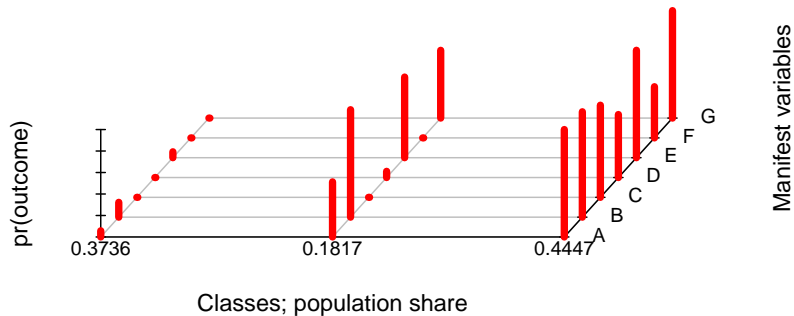
Figure 1: Estimation of the three-class basic latent class model using the `carcinoma` data; obtained by setting `graphs = TRUE` in the `poLCA` function call. Each group of red bars represents the conditional probabilities, by latent class, of being rated positively by each of the seven pathologists (labeled A through G). Taller bars correspond to conditional probabilities closer to 1 of a positive rating.

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`. 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.

The full output from the estimation of model `lc3` is given below. First, the estimated class-conditional response probabilities $\hat{\pi}_{jrk}$ are reported for pathologists A through G, with each row corresponding to a latent class, and each column corresponding to a diagnosis; negative in the first column, and positive in the second. Thus, for example, a slide belonging to the first ("negative") class has a 94% chance of being rated free from carcinoma by rater A, an 86% chance of the same from rater B, an 100% chance from rater C, and so forth.

Next, the output provides the estimated mixing proportions $\hat{p}_r$ corresponding to the share of observations belonging to each latent class. These are the same values that appear in Figure 1. An alternative method for determining the size of the latent classes is to assign each observation to a latent class on an individual basis according to its model posterior class membership probability. Values using this technique are reported directly below the estimated mixing proportions. Congruence between these two sets of population shares often indicates a good fit of the model to the data.

The next set of results simply reports the number of observations, the number of fully observed cases (for data sets with missing values and `na.rm = FALSE`), the number of estimated parameters, residual degrees of freedom, and maximum log-likelihood. It is always worth checking to ensure that the number of residual degrees of freedom is non-negative; `poLCA` will output a warning message if this is the case.

Finally, `poLCA` outputs a number of goodness of fit statistics as described in Section 2.4. For the `carcinoma` data, the minimum AIC and BIC criteria both indicate that the three-class model is most parsimonious: with two classes, the AIC is 664.5 and the BIC is 706.1; with three classes, the AIC decreases to 633.4 and the BIC decreases to 697.1; and with four classes, the AIC increases again to 641.6 and the BIC increases to 727.5.

```
Conditional item response (column) probabilities,
 by outcome variable, for each class (row)

$A
          Pr(1)  Pr(2)
class 1:  0.9427 0.0573
class 2:  0.4872 0.5128
class 3:  0.0000 1.0000

$B
          Pr(1)  Pr(2)
class 1:  0.8621 0.1379
class 2:  0.0000 1.0000
class 3:  0.0191 0.9809

$C
          Pr(1)  Pr(2)
class 1:  1.0000 0.0000
class 2:  1.0000 0.0000
class 3:  0.1425 0.8575

$D
          Pr(1)  Pr(2)
class 1:  1.0000 0.0000
class 2:  0.9424 0.0576
class 3:  0.4138 0.5862

$E
          Pr(1)  Pr(2)
class 1:  0.9449 0.0551
class 2:  0.2494 0.7506
class 3:  0.0000 1.0000

$F
          Pr(1)  Pr(2)
class 1:  1.0000 0.0000
class 2:  1.0000 0.0000
class 3:  0.5236 0.4764

$G
          Pr(1)  Pr(2)
class 1:  1.0000 0.0000
class 2:  0.3693 0.6307
class 3:  0.0000 1.0000

Estimated class population shares
 0.3736 0.1817 0.4447
```

```
Predicted class memberships (by modal posterior prob.)
 0.3729 0.1949 0.4322


=========================================================
Fit for 3 latent classes:
=========================================================
number of observations: 118
number of estimated parameters: 23
residual degrees of freedom: 95
maximum log-likelihood: -293.705

AIC(3): 633.41
BIC(3): 697.1357
G^2(3): 15.26171 (Likelihood ratio/deviance statistic)
X^2(3): 20.50336 (Chi-square goodness of fit)
```

5.2. 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.

Models with one covariate

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. 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. Next, estimate the latent class regression model and assign those results to object `nes.party`. A call to the `poLCA.reorder` command, with a subsequent re-estimation of the model, ensures that the three latent classes are assigned the same category labels in each run.

```
R> data("election")
R> f.party <- cbind(MORALG, CARESG, KNOWG, LEADG, DISHONG, INTELG,
+    MORALB, CARESB, KNOWB, LEADB, DISHONB, INTELB) ~ PARTY
```

```
R> nes.party <- poLCA(f.party, election, nclass = 3, verbose = FALSE)
R> probs.start <- poLCA.reorder(nes.party$probs.start,
+    order(nes.party$P, decreasing = TRUE))
R> nes.party <- poLCA(f.party, election, nclass = 3,
+    probs.start = probs.start)
```

By examining the estimated class-conditional response probabilities, we confirm that 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.

This example also illustrates a shortcoming of the χ^2 goodness of fit statistic, which is calculated to be over 34.5 billion. With only 1300 observations but nearly 17 million cells in the observed cross-classification table (that is, four responses to each of 12 questions, or 4^{12} cells), the vast majority of the cells will contain zero cases. For models such as this, using the χ^2 statistic to assess model fit is not advised.

In addition to the information outputted for the basic model, the poLCA output now also includes the estimated coefficients $\hat{\boldsymbol{\beta}}_r$ on the covariates, and their standard errors.

```
==========================================================
Fit for 3 latent classes:
==========================================================
2 / 1
          Coefficient  Std. error  t value  Pr(>|t|)
(Intercept)   -3.81813     0.31109  -12.274         0
PARTY          0.79327     0.06232   12.728         0
==========================================================
3 / 1
          Coefficient  Std. error  t value  Pr(>|t|)
(Intercept)    1.16155     0.17989    6.457         0
PARTY         -0.57436     0.06401   -8.973         0
==========================================================
```

Here, the neutral group is the first latent class, the favor-Bush group is the second latent class, and the favor-Gore group is the third latent class. Following the terminology in Section 3.1, the log-ratio prior probability that a respondent will belong to the favor-Bush group with respect to the neutral group is $\ln(p_{2i}/p_{1i}) = -3.82 + 0.79 \times$ PARTY. Likewise, the log-ratio prior probability that a respondent will belong to the favor-Gore group with respect to the neutral group is $\ln(p_{3i}/p_{1i}) = 1.16 - 0.57 \times$ PARTY. Equation 12 provides the formula for converting these log-ratios into predicted prior probabilities for each latent class.

To interpret the estimated generalized logit coefficients, we calculate and plot predicted values of p_{ri}, the prior probability of class membership, at varying levels of party ID. The R commands to do this are as follows, producing the graph in Figure 2.

```
R> pidmat <- cbind(1, c(1:7))
R> exb <- exp(pidmat %*% nes.party$coeff)
R> matplot(c(1:7), (cbind(1, exb)/(1 + rowSums(exb))),
+    main = "Party ID as a predictor of candidate affinity class",
```

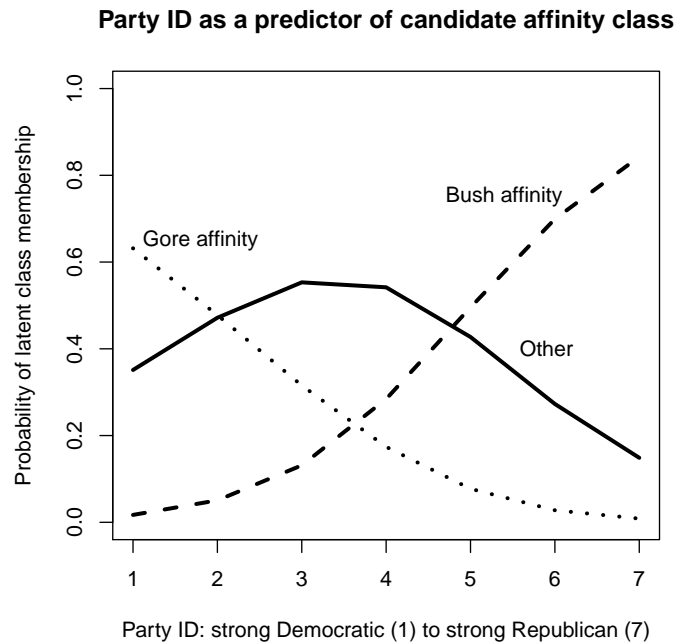**Party ID as a predictor of candidate affinity class**

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.

```
+     xlab = "Party ID: strong Democratic (1) to strong Republican (7)",
+     ylab = "Probability of latent class membership",
+     ylim = c(0, 1), type = "l", lwd = 3, col = 1)
R> text(5.9, 0.40, "Other")
R> text(5.4, 0.75, "Bush affinity")
R> text(1.8, 0.65, "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).

Models with more than one covariate

It is straightforward to similarly investigate models with more than one covariate. Suppose we are interested in whether the effect of age modifies the effect of partisanship on candidate affinity. We specify the interaction model with three covariates:

```
R> f.3cov <- cbind(MORALG, CARESG, KNOWG, LEADG, DISHONG, INTELG,
+     MORALB, CARESB, KNOWB, LEADB, DISHONB, INTELB) ~ PARTY * AGE
R> nes.3cov <- poLCA(f.3cov, election, nclass = 3, verbose = FALSE)
R> probs.start <- poLCA.reorder(nes.3cov$probs.start,
+     order(nes.3cov$P, decreasing = TRUE))
R> nes.3cov <- poLCA(f.3cov, election, nclass = 3,
+     probs.start = probs.start)
```

This produces the following coefficient estimates, again with the neutral group as the first latent class, the favor-Bush group as the second latent class, and the favor-Gore group as the third latent class.

```
==========================================================
Fit for 3 latent classes:
==========================================================
2 / 1
          Coefficient  Std. error  t value  Pr(>|t|)
(Intercept)   -4.39452     0.85423   -5.144     0.000
PARTY          0.80682     0.17614    4.581     0.000
AGE            0.01314     0.01772    0.741     0.459
PARTY:AGE     -0.00020     0.00363   -0.054     0.957
==========================================================
3 / 1
          Coefficient  Std. error  t value  Pr(>|t|)
(Intercept)   -0.31445     0.56324   -0.558     0.577
PARTY         -0.39923     0.19990   -1.997     0.046
AGE            0.02967     0.01121    2.648     0.008
PARTY:AGE     -0.00310     0.00398   -0.778     0.437
==========================================================
```

To see the effects of age on the candidate affinity of strong partisans, we first specify a matrix of hypothetical values of the covariates: `strdems` for Democrats and `strreps` for Republicans. We then calculate and plot the predicted prior probabilities of latent class membership corresponding to each of these chosen hypothetical values (Figure 3).

```
R> strdems <- cbind(1, 1, c(18:80), (c(18:80) * 1))
R> exb.strdems <- exp(strdems %*% nes.3cov$coeff)
R> matplot(c(18:80), (cbind(1, exb.strdems)/(1+rowSums(exb.strdems))),
+    main = "Age and candidate affinity for strong Democrats",
+    xlab = "Age", ylab = "Probability of latent class membership",
+    ylim = c(0, 1), type = "l", col = 1, lwd = 3)
R> strreps <- cbind(1, 7, c(18:80), (c(18:80) * 7))
R> exb.strreps <- exp(strreps %*% nes.3cov$coeff)
R> matplot(c(18:80), (cbind(1, exb.strreps) / (1 + rowSums(exb.strreps))),
+    main = "Age and candidate affinity for strong Republicans",
+    xlab = "Age", ylab = "Probability of latent class membership",
+    ylim = c(0, 1), type = "l", col = 1, lwd = 3)
```

As expected, regardless of age, strong Democrats are very unlikely to belong to the Bush-affinity group, and strong Republicans are very unlikely to belong to the Gore-affinity group. However, it is interesting to observe that while strong Republicans in 2000 had extremely high levels of affinity for Bush at all ages, strong Democrats below the age of 30 tended to be just as (or more) likely to belong to the neutral group as to the Gore-affinity group.
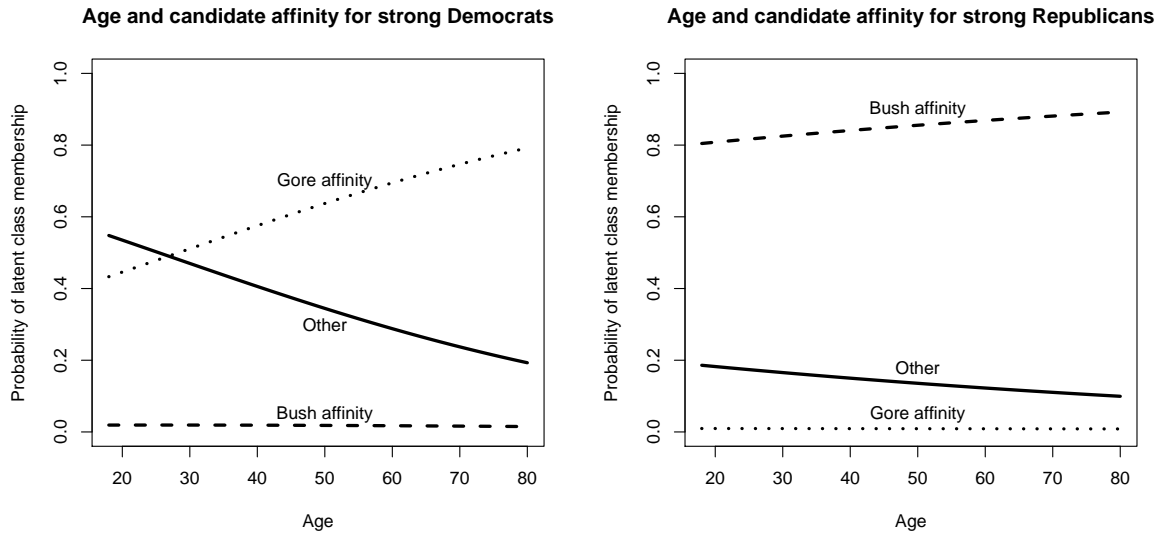
Age and candidate affinity for strong Democrats   **Age and candidate affinity for strong Republicans**

Figure 3: Predicted prior probabilities of latent class membership for strong Democrats (left) and strong Republicans (right) at ages 18-80.

6. Conclusion

The R package **poLCA** provides an easy-to-use framework for the estimation of latent class models and latent class regression models for the analysis of multivariate categorical data. The manifest variables may contain any number of possible (polytomous) outcomes. In the latent class regression model, covariates may be used to predict individual observations' latent class membership. **poLCA** also includes tools for visualizing model results, creating simulated multivariate categorical data sets with an unobserved latent categorical structure, and post-processing the model results to produce various other quantities of interest, including expected cell percentages and posterior probabilities of latent class membership for either observed or hypothetical cases.

poLCA is still undergoing active development. Planned extensions include flexibility to relax the assumption of local independence among selected manifest variables, explicit treatment of manifest variables as ordinal as well as nominal, incorporation of sampling weights, and accommodation of user-specified constraints on the class-conditional response probabilities π_{jrk} as a way to simplify models, achieve model identifiability, test substantive hypotheses, and analyze model fit. Such constraints might, for example, require selected response probabilities to be set equal to one another across different classes, across manifest variables within classes, or equal to fixed constant values, as in Goodman (1974). This extension would also permit the estimation of so-called "simultaneous" latent class models across multiple groups where the groups are already known (or at least theorized) to exist in the data (Clogg and Goodman 1986). The researcher would then include in the model a manifest variable measuring this known categorization, and specify that one of the grouping variable response probabilities be fixed at 1.0 in each class.

References

Agresti A (2002). *Categorical Data Analysis*. John Wiley & Sons, Hoboken.

Akaike H (1973). "Information Theory and an Extension of the Maximum Likelihood Principle." In B Petrov, F Csake (eds.), *Second International Symposium on Information Theory*, pp. 267–281. Akademiai Kiado, Budapest, Hungary.

Bandeen-Roche K, Miglioretti DL, Zeger SL, Rathouz PJ (1997). "Latent Variable Regression for Multiple Discrete Outcomes." *Journal of the American Statistical Association*, **92**(440), 1375–1386.

Beath K (2011). ***randomLCA***: *Random Effects Latent Class Analysis*. R package version 0.7-4, URL http://CRAN.R-project.org/package=randomLCA.

Blaydes L, Linzer DA (2008). "The Political Economy of Women's Support for Fundamentalist Islam." *World Politics*, **60**(4), 576–609.

Bolck A, Croon M, Hagenaars J (2004). "Estimating Latent Structure Models with Categorical Variables: One-Step Versus Three-Step Estimators." *Political Analysis*, **12**(1), 3–27.

Breen R (2000). "Why Is Support for Extreme Parties Underestimated by Surveys? A Latent Class Analysis." *British Journal of Political Science*, **30**(2), 375–382.

Clinton J, Jackman S, Rivers D (2004). "The Statistical Analysis of Roll Call Data." *American Political Science Review*, **98**(2), 355–370.

Clogg CC, Goodman LA (1986). "On Scaling Models Applied to Data from Several Groups." *Psychometrika*, **51**(1), 123–135.

Dayton CM (1998). *Latent Class Scaling Analysis*. Sage Publications, Thousand Oaks, CA.

Dayton CM, Macready GB (1988). "Concomitant-Variable Latent-Class Models." *Journal of the American Statistical Association*, **83**(401), 173–178.

Dempster AP, Laird NM, Rubin DB (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm." *Journal of the Royal Statistical Society B*, **39**, 1–38.

Dimitriadou E, Hornik K, Leisch F, Meyer D, Weingessel A (2011). ***e1071***: *Misc Functions of the Department of Statistics (e1071), TU Wien*. R package version 1.5-25, URL http://CRAN.R-project.org/package=e1071.

Duffy D (2010). ***gllm***: *Generalised log-linear model*. R package version 0.33. C code in emgllmfitter by Andreas Borg, URL http://CRAN.R-project.org/package=gllm.

Everitt BS (1984). *An Introduction to Latent Variable Models*. Chapman and Hall, London.

Everitt BS, Hand DJ (1981). *Finite Mixture Distributions*. Chapman and Hall, London.

Feick LF (1989). "Latent Class Analysis of Survey Questions that Include Don't Know Responses." *Public Opinion Quarterly*, **53**(4), 525–547.

Forster MR (2000). "Key Concepts in Model Selection: Performance and Generalizability." *Journal of Mathematical Psychology*, **44**, 205–231.

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

Goodman L (1974). "Exploratory Latent Structure Analysis using both Identifiable and Unidentifiable Models." *Biometrika*, **61**, 315–231.

Grün B, Leisch F (2008). "FlexMix Version 2: Finite Mixtures with Concomitant Variables and Varying and Constant Parameters." *Journal of Statistical Software*, **28**(4), 1–35. URL http://www.jstatsoft.org/v28/i04/.

Hagenaars JA, McCutcheon AL (eds.) (2002). *Applied Latent Class Analysis.* Cambridge University Press, Cambridge.

Hennig C (2010). **fpc**: *Flexible Procedures for Clustering.* R package version 2.0-3, URL http://CRAN.R-project.org/package=fpc.

Hill JL, Kriesi H (2001a). "Classification by Opinion-Changing Behavior: A Mixture Model Approach." *Political Analysis*, **9**(4), 301–324.

Hill JL, Kriesi H (2001b). "An Extension and Test of Converse's 'Black-and-White' Model of Response Stability." *American Political Science Review*, **95**(2), 397–413.

Lazarsfeld PF (1950). "The Logical and Mathematical Foundations of Latent Structure Analysis." In SA Stouffer (ed.), *Measurement and Prediction*, pp. 362–412. John Wiley & Sons, New York.

Leisch F (2004). "FlexMix: A General Framework for Finite Mixture Models and Latent Class Regression in R." *Journal of Statistical Software*, **11**(8), 1–18. URL http://www.jstatsoft.org/v11/i08/.

Lin TH, Dayton CM (1997). "Model Selection Information Criteria for Non-Nested Latent Class Models." *Journal of Educational and Behavioral Statistics*, **22**(3), 249–264.

Linzer DA (2011). "Reliable Inference in Highly Stratified Contingency Tables: Using Latent Class Models as Density Estimators." *Political Analysis*, **19**(2).

Linzer DA, Lewis J (2011). **poLCA**: *Polytomous Variable Latent Class Analysis.* R package version 1.3, URL http://CRAN.R-project.org/package=poLCA.

McCutcheon AL (1985). "A Latent Class Analysis of Tolerance for Nonconformity in the American Public." *The Public Opinion Quarterly*, **49**(5), 474–488.

McCutcheon AL (1987). *Latent Class Analysis.* Sage Publications, Newbury Park.

McLachlan G, Peel D (2000). *Finite Mixture Models.* John Wiley & Sons, New York.

McLachlan GJ, Krishnan T (1997). *The EM Algorithm and Extensions.* John Wiley & Sons, New York.

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

R Development Core Team (2010). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.

Schwartz G (1978). "Estimating the Dimension of a Model." *The Annals of Statistics*, **6**, 461–464.

The National Election Studies (2000). *The 2000 National Election Study dataset*. University of Michigan, Center for Political Studies, producer and distributor, Ann Arbor, MI. URL http://www.electionstudies.org/.

Affiliation:

Drew A. Linzer
Department of Political Science
Emory University
327 Tarbutton Hall
1555 Dickey Drive
Atlanta, GA 30322, United States of America
E-mail: dlinzer@emory.edu
URL: http://userwww.service.emory.edu/~dlinzer/

Jeffrey B. Lewis
Department of Political Science
University of California, Los Angeles
Box 951472
4289 Bunche Hall
Los Angeles, CA 90095, United States of America
E-mail: jblewis@polisci.ucla.edu
URL: http://www.sscnet.ucla.edu/polisci/faculty/lewis/