LCCR: An R Package for Inference on Latent Class Models for Capture-Recapture Data with Covariates

by Francesco Bartolucci and Antonio Forcina

Abstract A detailed description of the R package LCCR for the analysis of capture-recapture data relating to a closed population is provided. The data that can be analyzed consist of the full capture history for each sample unit and may possibly include individual covariates. The package allows the specification and estimation of latent class models in which the distribution of the capture history conditional on the latent class and covariates follows either a log-linear model in which bivariate interactions are allowed, or a logit model for the conditional probability of capture at each occasion given the previous capture history. Alternatively, the conditional distribution of each capture occasion can be formulated by a logit model that may account for the effect of previous capture occasions. Apart from the conditional distribution of the capture history, the covariates can also affect the distribution of the class weights. Estimation is based on the unconditional maximum likelihood method, suitably extended to account for the presence of covariates by including unit-specific weights, which are commonly used in empirical likelihood methods. The package also allows simulation of capture-recapture data from a specified model and the computation of the profile confidence interval for the population size. For illustration, we use a data set about meningitis cases in an Italian region.

1 Introduction

In the statistical literature on capture-recapture data relating to closed populations, see Böhning et al. (2018) for a recent overview, several models are now available, which represent a development of traditional models such as those illustrated in Otis et al. (1978). Here we focus on the typical case in which, for each unit captured at least once, the full capture history is available. In this case, the most recent model formulations can address both observed heterogeneity, accounted for by individual covariates, and unobserved heterogeneity, using random effects or latent variables having a continuous or discrete distribution. Moreover, maximum likelihood estimation may be formulated in two different ways; see Sanathanan (1972) for a fundamental contribution in this regard. The simplest method is based on maximizing the conditional likelihood function of the observed data given that sample units have been captured at least once. This function depends on the model parameters only, which are estimated prior to the population size. In contrast, the unconditional maximum likelihood (UML) method is based on a target function which is jointly maximized with respect to the population size and model parameters.

In this paper, we focus on a class of models that may address both observed and unobserved heterogeneity, by the inclusion of individual covariates and latent variables having a discrete distribution, and which are described in Bartolucci and Forcina (2024); this class of models is closely related to the one proposed in Bartolucci and Forcina (2006) and Bartolucci and Pennoni (2007). In particular, we describe the R package **LCCR** (Bartolucci and Forcina, 2025) that contains functions to formulate and estimate these models. Estimation is based on the UML method described in Bartolucci and Forcina (2024) that takes inspiration from the formulation based on empirical likelihood (Owen, 2001), developed by Liu et al. (2017). The functions in package **LCCR** may also be used to simulate data from an assumed model and to obtain a profile confidence interval for the population size, which has interesting properties with respect to standard confidence intervals based on the standard error for this parameter.

In R, other packages are available to deal with capture-recapture data. Here we mention two among the most popular packages which are available in CRAN and are related to the present package **LCCR**. Package **Rcapture** (Baillargeon and Rivest, 2007) provides functions to fit log-linear models for capture-recapture data for closed and open populations, together with tools for preliminary analysis of these data based on descriptive statistics and for model selection and diagnostics. For closed populations, in particular, the models considered may be specified in different ways to also include behavioral and heterogeneity effects (see also Rivest and Baillargeon, 2007), while parameter estimation is based on a conditional maximum likelihood method. Inference on the population size is also based on the profile likelihood method of Cormack (1992). Another R package that may be used to fit capture-recapture models for closed populations is **BBRecapture**. The models considered in the package allow for accounting for behavioral effects in a flexible way; see also Alunni Fegatelli and Tardella (2016). The inference on these models is based on a Bayesian approach under a set of prior distributions on the parameters, although the package also contains functions for UML estimation. A related package is **LCMCR** (Manrique-Vallier, 2023) that, by a Markov chain Monte Carlo algorithm, fits a Bayesian non-parametric latent class model for capture-recapture data that does not require specifying the number of classes because it relies on a Dirichlet process, as described in Manrique-Vallier (2016). It is worth noting that, differently from the previous packages, the **LCCR** package described here allows for individual covariates conditionally on the latent class, apart from permitting the formulation of models with a variety of effects of the type conceptualized by Otis et al. (1978).

The remainder of the paper is organized as follows. In the next section, we provide an overview of the statistical approach in terms of model assumptions and inferential methods, summarizing some concepts introduced in Bartolucci and Forcina (2024). Then, we describe a set of preliminary functions of package **LCCR** to formulate models, simulate data from the assumed model, and manipulate the available data. A further section is devoted to the main R functions to estimate model parameters and the population size, also by a profile confidence interval. The paper concludes with an example based on Italian meningitis data and a brief summary.

2 Approach overview

Let J denote the number of capture occasions generating configurations denoted by $\boldsymbol{r} = (r_1, \ldots, r_J)'$, with $r_j = 0, 1$, $j \in \mathcal{J}$, where $\mathcal{J} = \{1, \ldots, J\}$. The set of all possible capture configurations $\boldsymbol{r}$ apart from $\mathbf{0}$ is denoted by $\mathcal{R}$, where in general $\mathbf{0}$ denotes a column vector of zeros of a suitable dimension; when this set includes the $\mathbf{0}$ configuration, it is denoted by $\tilde{\mathcal{R}}$. In the present framework, the units captured at least once, whose number is denoted by n, may be collected in I strata with each stratum $i \in \mathcal{I}$, $\mathcal{I} = \{1, \ldots, I\}$, having a corresponding vector of c covariates $\boldsymbol{w}_i$. We also consider the possible presence of vectors of covariates $\boldsymbol{x}_{i,j}$, of dimension d, which are also specific to the capture occasion. The observed capture-recapture configurations are represented by the column vectors $\boldsymbol{y}_i$ having $2^J - 1$ elements $y_{i,\boldsymbol{r}}$, $\boldsymbol{r} \in \mathcal{R}$, corresponding to the frequency of the capture configuration $\boldsymbol{r}$ in stratum i. The sum of these frequencies is denoted by n_i, with all the n_i collected in the vector $\boldsymbol{n}$. Note that, with individual data, each vector $\boldsymbol{y}_i$ has only one element equal to 1 and all other elements equal to 0, so that $n_i = 1$ for all i.

2.1 Model assumptions

The approach proposed in Bartolucci and Forcina (2024) is based on a latent class (LC) model with H classes and in which the class weights $\pi_{h,i}$, $h \in \mathcal{H}$, $i \in \mathcal{I}$, with $\mathcal{H} = \{1, \ldots, H\}$, may depend on the vectors of covariates $\boldsymbol{w}_i$. The class weights for the same stratum i are collected in the column vector $\boldsymbol{\pi}_i$. Moreover, the conditional probability of the capture configuration $\boldsymbol{r}$ given the latent class h and that the individual is in stratum i, denoted by $q_{h,i,\boldsymbol{r}}$, may depend on the vectors of covariates $\boldsymbol{x}_{i,j}$. These probabilities are collected in the column vector $\boldsymbol{q}_{h,i}$ for all $\boldsymbol{r} \in \mathcal{R}$ in lexicographical order. When also the capture-recapture configuration $\mathbf{0}$ is included, the probability vector at issue will be denoted by $\tilde{\boldsymbol{q}}_{h,i}$ that then has elements $q_{h,i,\boldsymbol{r}}$ for all $\boldsymbol{r} \in \tilde{\mathcal{R}}$. Two different approaches are adopted to model this vector of probabilities:

the first is based on a log-linear parametrization and the second is based on a recursive logit parametrization.

Finally, we recall that the probability of capture configuration $\boldsymbol{r}$ for stratum h is equal to

$$p_{i,\boldsymbol{r}} = \sum_{h\in\mathcal{H}} \pi_{h,i} q_{h,i,\boldsymbol{r}};$$

a related expression may be used to compute the overall vector of such probabilities, $\boldsymbol{p}_i$ (or $\tilde{\boldsymbol{p}}_i$ when also the elements corresponding to $\boldsymbol{r} = \boldsymbol{0}$ are included), on the basis of vectors $\boldsymbol{q}_{h,i}$ (or $\tilde{\boldsymbol{q}}_{h,i}$) and $\boldsymbol{\pi}_i$.

Before describing in detail the single model components, it is worth recalling that the LC approach to dealing with unobserved heterogeneity is rather common in the capture-recapture literature. Among the most recent papers dealing with this approach, that by Aleshin-Guendel et al. (2024) focuses on identifiability issues, although with reference to simpler models than the ones presented here.

Distribution of the latent classes

For the conditional distribution of the latent classes given the covariates, we assume a multinomial logit model based on the assumption that

$$\pi_{h,i} = \frac{1}{1+\sum_{l\in\tilde{\mathcal{H}}}\exp(\alpha_l + \boldsymbol{w}_i'\boldsymbol{\beta}_l)} \begin{cases} 1, & h = 1, \\ \exp(\alpha_h + \boldsymbol{w}_i'\boldsymbol{\beta}_h), & h \in \bar{\mathcal{H}}, \end{cases}$$

where $\bar{\mathcal{H}} = \{2, \ldots, H\}$. Using matrix notation, the overall vector of class weights may be expressed as

$$\boldsymbol{\pi}_i = \frac{\exp(\boldsymbol{W}_i\boldsymbol{\beta})}{\boldsymbol{1}'\exp(\boldsymbol{W}_i\boldsymbol{\beta})},$$

for a suitable design matrix $\boldsymbol{W}_i$ depending on $\boldsymbol{w}_i$ and where $\boldsymbol{1}$ denotes a column vector of ones of a suitable dimension, while $\boldsymbol{\beta}$ is the vector of all previous parameters.

Conditional response probabilities

The user can choose between two ways of modeling capture probabilities conditionally on the latent class. The first model formulation is based on a log-linear parametrization and assumes that

$$q_{h,i,\boldsymbol{r}} = \frac{\exp\left(\sum_{j\in\mathcal{J}} r_j\gamma_{h,j} + \sum_{j\in\mathcal{J}} r_j\boldsymbol{x}_{i,j}'\boldsymbol{\delta}_{h,j} + \sum_{(j_1,j_2)\in\mathcal{B}} r_{j_1}r_{j_2}\eta_{h,j_1,j_2}\right)}{\sum_{\boldsymbol{u}\in\tilde{\mathcal{R}}}\exp\left(\sum_{j\in\mathcal{J}} u_j\gamma_{h,j} + \sum_{j\in\mathcal{J}} u_j\boldsymbol{x}_{i,j}'\boldsymbol{\delta}_{h,j} + \sum_{(j_1,j_2)\in\mathcal{B}} u_{j_1}u_{j_2}\eta_{h,j_1,j_2}\right)}, \tag{1}$$

where $\gamma_{h,j}$ are main effects, $\boldsymbol{\delta}_{h,j}$ are regression coefficients for the covariates in each vector $\boldsymbol{x}_{i,j}$, and η_{h,j_1,j_2} are bivariate interactions. Moreover, $\mathcal{B}$ is a set of pairs of indices of type (j_1, j_2) of the bivariate interactions, which is possibly empty when these interactions are not used and then the third sum at the numerator and denominator of equation (1) disappears. The number of these interactions is denoted by $b = |\mathcal{B}|$. Obviously, when covariates are not included, even the second sum disappears from the previous equation. All the parameters in (1) are collected in the vector $\tilde{\boldsymbol{\lambda}}$ and suitable constraints, corresponding to the linear form $\tilde{\boldsymbol{\lambda}} = \boldsymbol{L}\boldsymbol{\lambda}$ for a suitable matrix $\boldsymbol{L}$, can be defined in order to make the model more parsimonious. The types of constraint will be discussed in the following when the package is illustrated.

For the overall vector of probabilities we have

$$\tilde{\boldsymbol{q}}_{h,i} = \frac{\exp(\boldsymbol{M}_{h,i}\boldsymbol{\lambda})}{\boldsymbol{1}'\exp(\boldsymbol{M}_{h,i}\boldsymbol{\lambda})}, \tag{2}$$

based on design matrices $\boldsymbol{M}_{h,i}$ depending on $\boldsymbol{L}$ and having a number of rows equal to 2^J and a number of columns equal to that of parameters in (1), which are collected in $\boldsymbol{\lambda}$. In practice, $\boldsymbol{M}_{h,i}\boldsymbol{\lambda}$ contains the terms of the exponential function at the numerator of (1) for all $\boldsymbol{r} \in \tilde{\mathcal{R}}$.

The second model formulation is based on a recursive logit model according to which

$$q_{h,i,\boldsymbol{r}} = \frac{\exp[r_j(\gamma_{h,1} + \boldsymbol{x}'_{i,j}\boldsymbol{\delta}_{h,1})]}{1+\exp(\gamma_{h,1} + \boldsymbol{x}'_{i,j}\boldsymbol{\delta}_{h,1})} \prod_{j=2}^{J} \frac{\exp[r_j(\gamma_{h,j} + \boldsymbol{x}'_{i,j}\boldsymbol{\delta}_{h,j} + f(\boldsymbol{r}_{j-1})\eta_{h,j})]}{1+\exp(\gamma_{h,j} + \boldsymbol{x}'_{i,j}\boldsymbol{\delta}_{h,j} + f(\boldsymbol{r}_{j-1})\eta_{h,j})}, \tag{3}$$

where $f(\boldsymbol{r}_{j-1})$ is a function of the past capture history. In particular, we consider the case in which this function is equal to $I(\boldsymbol{1}'\boldsymbol{r}_{j-1} > 0)$, so that it is equal to 1 if the individual has already been marked and 0 otherwise, or simply $f(\boldsymbol{r}_{j-1}) = \boldsymbol{1}'\boldsymbol{r}_{j-1}$. We also consider the case $f(\boldsymbol{r}_{j-1}) = r_{j-1}$, so that only the previous response variable is relevant. Also in this case, we can assume specific constraints that will be described in the following. The parameters in (3) are still collected in the vector $\tilde{\boldsymbol{\lambda}}$ on which the constraint $\tilde{\boldsymbol{\lambda}} = \boldsymbol{L}\boldsymbol{\lambda}$ is assumed, so that the free parameter vector is $\boldsymbol{\lambda}$.

A convenient way to express this parameterization is as

$$\tilde{\boldsymbol{q}}_{h,i} = \exp\{\boldsymbol{A}\boldsymbol{M}_{h,i}\boldsymbol{\lambda} - \boldsymbol{B}\log[\boldsymbol{1} + \exp(\boldsymbol{M}_{h,i}\boldsymbol{\lambda})]\}, \tag{4}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ are suitable matrices having dimension $2^J \times (2^J - 1)$ and with all elements equal to 0 or 1. Moreover, the $\boldsymbol{M}_{h,i}$ are design matrices such that $\boldsymbol{M}_{h,i}\boldsymbol{\lambda}$ is a vector containing the logits corresponding to the single terms in (3).

2.2 Unconditional maximum likelihood inference

Inference on the models described above follows the approach proposed by Liu et al. (2017) and further developed in Bartolucci and Forcina (2024). In summary, we associate a weight τ_i to each stratum i under the constraint that $\sum_{i\in\mathcal{I}} \tau_i = 1$, so that the probability of not being captured is equal to

$$\phi = \sum_{i\in\mathcal{I}} \tau_i \phi_i,$$

where $\phi_i = p_{i,\boldsymbol{0}}$. All such probabilities will be collected in vector $\boldsymbol{\phi}$.

We base inference on the log-likelihood function

$$\ell(\boldsymbol{\psi}) = \log \frac{\Gamma(N+1)}{\Gamma(N-n+1)} + (N-n)\log(\phi) + \sum_{i\in\mathcal{I}} (\boldsymbol{y}_i' \log \boldsymbol{p}_i + \log \tau_i),$$

where $\boldsymbol{\psi}$ denotes the overall vector of parameters including N. In order to maximize this function, we rely on an algorithm that is based on the following steps:

1. for fixed $\boldsymbol{\tau}$ and N, maximize $\ell(\boldsymbol{\psi})$ with respect to $\boldsymbol{\beta}$ and $\boldsymbol{\lambda}$. This amounts to maximizing

$$\ell_1(\boldsymbol{\beta}, \boldsymbol{\lambda}) = (N-n)\log(\phi) + \sum_{i=1}^{s} \boldsymbol{y}_i' \log(\boldsymbol{p}_i) \tag{5}$$

   with respect to these parameters, as will be clarified below;

2. for fixed $\boldsymbol{\beta}$, $\boldsymbol{\lambda}$, and N, maximize $\ell(\boldsymbol{\psi})$ with respect to $\boldsymbol{\tau}$. This amounts to maximizing

$$\ell_2(\boldsymbol{\tau}) = (N-n)\log(\phi) + \sum_{i=1}^{s} \log \tau_i$$

   and may be based on a simple updating rule consisting in computing

$$\boldsymbol{\tau} = \frac{1}{N}\left[\boldsymbol{n} + \frac{N-n}{\phi}\mathrm{diag}(\boldsymbol{\phi})\boldsymbol{\tau}\right],$$

where the previous $\boldsymbol{\tau}$ and ϕ are included at the right-hand side; see Bartolucci and Forcina (2024) for details;

3. for fixed $\boldsymbol{\beta}$, $\boldsymbol{\lambda}$, and $\boldsymbol{\tau}$, maximize $\ell(\boldsymbol{\psi})$ with respect to N. This amounts to maximizing

$$\ell_3(N) = \log \frac{\Gamma(N+1)}{\Gamma(N-n+1)} + (N-n)\log(\phi)$$

with respect to N, which may be accomplished by simple Newton-Raphson steps.

Maximization of $\ell_1(\boldsymbol{\beta}, \boldsymbol{\lambda})$ in (5) is based on an Expectation-Maximization (EM) algorithm (Dempster et al., 1977) that has a different implementation according to whether a log-linear or a recursive logit parametrization is adopted, while in Bartolucci and Forcina (2024) a Fisher-scoring maximization algorithm is suggested. The EM algorithm alternates two steps until convergence:

- **E-step**: for each stratum i, the expected value of the missing frequency $y_{i,\mathbf{0}}$ is computed given the current value of the parameters and the observed data as

$$\hat{y}_{i,\mathbf{0}} = (N-n)\frac{\phi_i \tau_i}{\phi}.$$

Then, all observed and predicted frequencies are split among classes, obtaining

$$\tilde{y}_{h,i,\boldsymbol{r}} = y_{i,\boldsymbol{r}} \frac{\pi_{h,i} q_{h,i,\boldsymbol{r}}}{p_{i,\boldsymbol{r}}}, \quad h \in \mathcal{H},\ i \in \mathcal{I},\ \boldsymbol{r} \in \tilde{\mathcal{R}},$$

with $y_{i,\mathbf{0}} \equiv \hat{y}_{i,\mathbf{0}}$, so that $\hat{N}_i = \hat{y}_{i,\mathbf{0}} + n_i$ is the predicted size for this stratum at the population level.

- **M-step**: the parameters $\boldsymbol{\beta}$ are updated by maximizing the log-likelihood

$$\tilde{\ell}_1(\boldsymbol{\beta}) = \sum_{i \in \mathcal{I}} \tilde{\boldsymbol{n}}_i' \log \boldsymbol{\pi}_i,$$

where $\tilde{\boldsymbol{n}}_i$ is the column vector with elements $\tilde{n}_{h,i} = \sum_{\boldsymbol{r}} \tilde{y}_{h,i,\boldsymbol{r}}$, $h = 1, \ldots, H$. Within this step, the parameters $\boldsymbol{\lambda}$ are updated by maximizing

$$\tilde{\ell}_2(\boldsymbol{\lambda}) = \sum_{h \in \mathcal{H}} \sum_{i \in \mathcal{I}} \tilde{\boldsymbol{y}}_{h,i}' \log \tilde{\boldsymbol{q}}_{h,i},$$

where $\tilde{\boldsymbol{y}}_{h,i}$ is the column vector with elements $\tilde{y}_{h,i,\boldsymbol{r}}$ for all $\boldsymbol{r} \in \tilde{\mathcal{R}}$. The derivatives of functions $\tilde{\ell}_1(\boldsymbol{\beta})$ and $\tilde{\ell}_2(\boldsymbol{\lambda})$ are provided in Appendix depending on the type of model adopted.

The EM algorithm must be properly initialized in order to increase the chance of getting the global maximum of the overall log-likelihood function. As will be illustrated in the following, this may be based on the joint use of deterministic and random starting values; more details are provided in Appendix. We also warn the reader that, as is well known, the EM algorithm may require many iterations as a counterpart to its stability.

Within package **LCCR**, the estimation function also computes the standard errors for the parameter estimates, including the population size N. This is based on first obtaining the observed information matrix as minus the numerical derivative of the score vector with respect to all model parameters, with the exception of parameters $\boldsymbol{\tau}$ that are updated on the basis of the value of the other parameters and are treated as nuisance parameters. Then, this information matrix is dealt with in the usual way to obtain the standard errors. We refer to Bartolucci and Forcina (2024) for details about the computation of the score.

3 Preliminary functions in the package

The approach discussed in this paper is based on a variety of parametrizations regarding the conditional response probabilities given the latent class. Before illustrating the functions in package **LCCR** that may be used for estimation, it is then important to illustrate how to specify the model of interest and the functions that build the design matrices used to define the log-linear and recursive logit parametrizations in (2) and (4). We also illustrate other functions for data simulation and for data manipulation. In this regard, it is important to recall that we consider two possible data formats. The first is adopted with individual data, so that $n_i = 1$, $i \in \mathcal{I}$, and then the observed capture configurations may be equivalently represented by the vectors $\boldsymbol{r}_i$ specified for all individuals. The second format is with aggregated data in which every n_i may be greater than 1 and the observed capture configurations are represented by the frequency vectors $\boldsymbol{y}_i$.

3.1 Model specification

We consider first the log-linear parametrization. The model is formulated by specifying the constraints regarding the intercepts $\gamma_{h,j}$, the vector of regression coefficients $\delta_{h,j}$, and the bivariate interaction parameters η_{h,j_1,j_2}. The possible constraints are listed below.

1. Regarding the intercepts $\gamma_{h,j}$, it is possible to formulate the following assumptions:
   - standard LC model with a separate intercept $\gamma_{h,j}$ for each class and capture occasion (overall HJ parameters);
   - an LC model with the same intercept for each capture occasion, so that $\gamma_{h,j}$ does not depend on j (overall H parameters);
   - a Rasch LC model in which $\gamma_{h,j}$ is decomposed as the sum of a parameter for each latent class, measuring the tendency to be captured, and a parameter for the capture occasion (overall $H-1+J$ parameters taking identifiability constraints into account).
2. Regarding the regression coefficients, the following restrictions are considered:
   - same regression coefficients across classes and capture occasions, that is, $\delta_{h,j}$ does not depend either on h or j (overall d parameters, where d is the number of covariates in $\boldsymbol{x}_{i,j}$);
   - regression coefficients that are class specific, that is, $\delta_{h,j}$ does not depend on j (overall dH parameters);
   - regression coefficients that are occasion specific, that is, $\delta_{h,j}$ does not depend on h (overall dJ parameters);
   - free regression coefficients for each latent class and capture occasion (overall dHJ parameters).
3. Regarding the bivariate interactions, the possible constraints are:
   - same interaction for all latent classes, that is, η_{h,j_1,j_2} is constant with respect to j_1 and j_2 (overall 1 parameter);
   - interactions that are latent class specific, that is, η_{h,j_1,j_2} depends only on h (overall H parameters);
   - interactions that are specific for each element in $\mathcal{B}$, that is, η_{h,j_1,j_2} does not depend on h (overall b parameters);
   - free interactions η_{h,j_1,j_2} (overall bH parameters).

The function that builds matrices $\boldsymbol{M}_{h,i}$ in (2) depending on the specified model is

```
design_matrix_loglin(J, H = 1, main = c("LC", "same", "Rasch"), X = NULL,
                     free_cov = c("no", "class", "resp", "both"),
                     biv = NULL, free_biv = c("no", "class", "int", "both"))
```

with the following input arguments:

- J: number of capture occasions;
- H: number of latent classes;
- main to specify the constraints on the intercepts with possible values: LC for the LC specification; same for the same effect for each capture occasion; and Rasch for additive effect of class and capture occasion;
- X: array containing covariate vectors $\boldsymbol{x}_{h,i}$ with dimension $S \times d \times J$;
- free_cov to specify the constraints on the regression parameters $\delta_{h,i}$ with possible values: no for the constant effect with respect to class and capture occasion; class for free effects with respect to the class; resp for free effects with respect to the capture occasion; both for free effect with respect to the class and capture occasion;
- biv: matrix of dimension $b \times 2$ containing the list of bivariate interactions;
- free_biv to specify the constraint on the interaction parameters δ_{h,j_1,j_2} with possible values: no for constant effect with respect to the class and interaction; class for free effect with respect to the class; int for free effect with respect to interaction; and both for free effects with respect to the class and interaction.

Just to clarify the use of this function, suppose that there are 3 capture occasions and a single group covariate equal to 0 and 1. Then, by the following commands we can obtain the design matrices for an LC model with two classes and the effect of the covariate:

```
X = array(c(0,1),c(2,1,3))
M = design_matrix_loglin(3,2,X=X)$M
```

In this way we obtain an array M having dimension $8 \times 7 \times 2 \times 2$, where 8 is the number of response configurations, 7 is the number of parameters involved in the conditional distribution of the capture configurations given the latent class, and 2 is both the number of strata and classes.

Regarding the logit recursive model parameters in (3), we may assume similar constraints as above for the parameters $\delta_{h,j}$ and $\eta_{h,j}$. Regarding these parameters, we may assume:

- the same parameter for all latent classes, that is, $\delta_{h,j}$ is constant with respect to h and j (overall d parameters);
- lagged effect that is latent class specific, that is, $\eta_{h,j}$ is independent of j (overall H parameters);
- lagged effect that is capture occasion specific, that is, $\eta_{h,j} = \eta_j$ depends only on j (overall $J - 1$ parameters);
- free lagged effect parameters $\eta_{h,j}$ (overall $H(J - 1)$ parameters).

In this way, we also properly account for behavioral effects; see also Alunni Fegatelli and Tardella (2016).

Regarding the covariates, it is important to note that when there is only one covariate affecting the class weights, the I values of this covariate in the input argument X can be given as a vector, a matrix, or an array, whereas when there are more covariates X can be a matrix or an array. When X is a vector rather than an array, then it is interpreted as containing the values of a single covariate that are replicated for all capture occasions. Similarly, when X is a matrix of dimension $I \times d$, then its rows are replicated for all capture occasions.

The function in the package that builds the design matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{M}_{h,i}$ used in (4) is the following:

```
design_matrix_logit(J, H = 1, main = c("LC", "same", "Rasch"), X = NULL,
                    free_cov = c("no", "class", "resp", "both"),
                    flag = c("no", "prev", "sum", "atleast"),
                    free_flag = c("no", "class", "resp", "both"))
```

with the same input arguments as function `design_matrix_loglin()` defined above, apart from the following:

- `flag` to specify the dependence on the lagged responses: `no` for absence of dependence; `prev` for dependence on the previous response; `sum` for dependence on the sum of the previous response variables; and `atleast` to introduce a dummy variable equal to 1 if there is at least one capture in the past;
- `free_flag` to specify the constraints on the parameters for the lagged responses with values: `no` for using only one parameter; `class` when these parameters are free with respect to the latent class; `resp` when they are free with respect to the capture occasion; and `both` when they are free with respect to the class and capture occasion.

3.2 Simulation

The following function may be used to simulate data from one of the models formulated in the section about model assumptions:

```
simLCCR(H, J, be, la, N, model = c("loglin", "logit"), Wc = NULL, Xc = NULL, biv = NULL,
        flag = c("no", "prev", "sum", "atleast"),
        main = c("LC", "same", "Rasch"),
        free_cov = c("no", "class", "resp", "both"),
        free_biv = c("no", "class", "int", "both"),
        free_flag = c("no", "class", "resp", "both"))
```

The input arguments are the following:

- `H`: number of latent classes;
- `J`: number of capture occasions;
- `be`: parameter vector on the class weights;
- `la`: parameter vector on the conditional response probabilities;
- `N`: population size (with individual data) or vector of the size of every stratum in the population (with aggregated data);
- `model`: to specify the model formulation with options `loglin` or `logit`;
- `Wc`: matrix with rows corresponding to the vectors $\boldsymbol{w}_i$ for all sampled individuals;
- `Xc`: array of covariates in $\boldsymbol{x}_{h,i}$ organized as in the input of functions `design_matrix_loglin()` and `design_matrix_logit()` for all sampled individuals;
- `biv`, `main`, `free_cov`, `free_biv`: see the same arguments of function `design_matrix_loglin()` when the log-linear parametrization is adopted;
- `flag`, `main`, `free_cov`, `free_flag`: see the same arguments of function `design_matrix_logit()` when the recursive logit parametrization is adopted.

3.3 Data manipulation

Typically, individual data are available in the form of vectors $\boldsymbol{r}_i$ collected in the matrix $\boldsymbol{R}$. To use the estimation function that will be illustrated below, it is then necessary to transform each of these capture configurations into a frequency vector of type $\boldsymbol{y}_i$ having dimension 2^J. For this aim, with package LCCR it is possible to use function

```
freq_data(R,count=rep(1,nrow(R)))
```

that accepts as input the matrix of individual capture configurations and, as an optional argument, a vector of counts for each of these configurations, and provides the matrix of the corresponding frequencies.

Furthermore, it may be convenient to transform individual data into aggregated data, and for this aim it is possible to use function

```
aggr_data(Y,W=NULL,X=NULL)
```

with the following input arguments:

- Y: matrix of capture configurations in frequency format;
- W: matrix of covariates affecting the class weights;
- X: array of covariates affecting the conditional response probabilities given the latent class.

The same arguments are provided in output in aggregated form and called as Ya, Wa, and Xa.

4 Model estimation within the package

4.1 Point estimation

Within the package, the estimation function is

```
estLCCR(Y, H, model = c("loglin", "logit"), W = NULL, X = NULL, N = NULL, biv = NULL,
        flag = c("no", "prev", "sum", "atleast"),
        main = c("LC", "same", "Rasch"),
        free_cov = c("no", "class", "resp", "both"),
        free_biv = c("no", "class", "int", "both"),
        free_flag = c("no", "class", "resp", "both"),
        N0 = NULL, be0 = NULL, la0 = NULL, control = list(),
        verb = TRUE, init_rand = FALSE, se_out = FALSE)
```

The main input arguments are:

- Y: matrix of dimension $I \times (2^J - 1)$ with rows corresponding to the observed vectors $\boldsymbol{y}_i$;
- H: number of latent classes;
- model: to specify the type of parametrization of the capture probabilities given the latent class;
- W: matrix with rows corresponding to the vectors $\boldsymbol{w}_i$;
- X: array of covariates in $\boldsymbol{x}_{h,i}$ organized as in the input of functions design_matrix_loglin() and design_matrix_logit();
- N: fixed population size;

- `biv`, `main`, `free_cov`, `free_biv`: see the same arguments of function `design_matrix_loglin()` when the log-linear parametrization is adopted;
- `flag`, `main`, `free_cov`, `free_flag`: see the same arguments of function `design_matrix_logit()` when the recursive logit parametrization is adopted.

The main output arguments are:

- `be`: estimate of the parameters in $\boldsymbol{\beta}$;
- `la`: estimate of the parameters in $\boldsymbol{\lambda}$;
- `lk`: final log-likelihood;
- `N`: estimate of N;
- `AIC`: value of the Akaike Information Criterion (AIC) for model selection Akaike (1973);
- `BIC`: value of the Bayesian Information Criterion (BIC) for model selection Schwarz (1978);
- `tauv`: estimate of the vector of weights of each stratum;
- `phiv`: estimate of the vector probabilities of being never captured for each stratum;
- `Piv`: matrix of the probabilities $\pi_{h,i}$ of the latent classes for each stratum;
- `Q`: array of the conditional probabilities $q_{h,i,\boldsymbol{r}}$ of the capture configurations given each latent class and stratum;
- `seN`: standard error for the estimate of N;
- `sebe`: vector of standard errors for the estimate of $\boldsymbol{\beta}$;
- `sela`: vector of standard errors for the estimate of $\boldsymbol{\lambda}$.

4.2 Confidence interval on N

One interesting feature of package **LCCR** is the possibility to obtain a profile confidence interval for the population size that is based on the asymptotic results proposed in Liu et al. (2017) and Bartolucci and Forcina (2024). For this aim, it is possible to use the method

```
confint(object, parm = list(), level = 0.95, ...)
```

where:

- `object`: output from function `estLCCR`;
- `parm`: a list containing control arguments for the step length of the N values, range of N values in terms of distance of the log-likelihood from its maximum, and maximum value of this grid as a multiple of the estimate of this parameter;
- level: required confidence level.

The output of this function can be suitably represented as will be shown in the following section about an illustrative example. We warn the user that, given the approach followed to obtain confidence intervals, this function could require a long computing time. However, by suitably setting the arguments in `parm`, it is possible to speed up the process to obtain the confidence interval.

5 Example about victims of trafficking

For the first illustration of the package, which is more focused on data preparation, we make use of data taken from Silverman (2020) about victims of trafficking in the UK in 2013; see also Silverman (2014) for details. Five lists are considered: *LA* (local authorities); NG (non-government organizations); *PFNCA* (police forces and National Crime Agency); *GO* (government organizations); *GP* (general public).

In the format reporting the frequency of every observed capture configuration $\boldsymbol{r}$, the data can be entered as follows:

```
UKdat5 = data.frame(LA = c(1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1),
                    NG = c(0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1),
                 PFNCA = c(0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1),
                    GO = c(0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1),
                    GP = c(0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0),
                 count = c(54, 463, 995, 695, 316, 15, 19, 3, 62, 19,
                           1, 76, 11, 8, 1, 1, 4, 1))
```

Note that the resulting data frame has 18 rows, corresponding to the number of distinct configurations, and 6 columns, as the last column is that of the frequencies. In order to convert these data into the format used in package **LCCR**, based on vectors of type $\boldsymbol{y}_i$ having as elements the frequencies of all possible capture configurations for stratum i, we can proceed as follows:

```
# load package
require(LCCR)

# prepare data
Y = freq_data(as.matrix(UKdat5[,1:5]),UKdat5[,6])
Ya = aggr_data(Y)$Ya
```

In this way we obtain the following matrix of a single row, as there is only one stratum, and 32 columns equal to the number of capture configurations, that is, 2^5:

```
> Ya
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14] [,15] [,16]
[1,]    0  316  695    8  995   11   76    0  463     1    19     0    62     0     4     0
     [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25] [,26] [,27] [,28] [,29] [,30]
[1,]    54     0     3     0    19     0     0     0    15     0     1     0     1     0
     [,31] [,32]
[1,]     1     0
```

These configurations are arranged in lexicographical order, and then, for instance, 316 is the frequency of the second configuration equal to $\boldsymbol{r} = (0,0,0,0,1)'$.

To fit the simple independence model M_t on these data, we can use the following command:

```
est1 = estLCCR(Ya,H=1,se_out=TRUE)
```

obtaining the following output:

```
> summary(est1)

Estimation of latent class models for capture-recapture data

Call:
estLCCR(Y = Ya, H = 1, se_out = TRUE)
```

```
Available objects:
 [1] "beta"      "lambda"    "lk"        "N"         "np"        "AIC"
 [7] "BIC"       "M"         "tauv"      "phiv"      "Piv"       "Q"
[13] "lk1"       "lk2"       "lk3"       "lk4"       "call"      "Y"
[19] "H"         "model"     "W"         "X"         "biv"       "flag"
[25] "main"      "free_cov"  "free_biv"  "free_flag" "se_out"    "seN"
[31] "selambda"

   LogLik        np       AIC       BIC
 14278.46      5.00 -28546.93 -28517.34

Population size:
      est.     s.e.
N 13435.07 805.6171

Parameters affecting the conditional capture probabilities given the latent class:
           est.       s.e.    t-test p-value
main1 -4.955289 0.11983134 -41.35219       0
main2 -3.122125 0.07590029 -41.13456       0
main3 -2.350668 0.07246115 -32.44038       0
main4 -2.750334 0.07340456 -37.46816       0
main5 -3.663166 0.08267378 -44.30868       0
```

The output contains the maximum likelihood at convergence, number of parameters, and AIC and BIC values. Moreover, it reports the estimate of the population size, with standard error, and each probability of capture on the logit scale together with other statistics.

To estimate, on the same data, a simple model with two classes, it is possible to use the command

```
est2 = estLCCR(Ya,H=2,se_out=TRUE)
```

The output of the function can be obtained by the method `summary()` as usual. This output will be illustrated in more detail in the following example about meningitis data. Here, it is important to note that the EM algorithm requires a large number of iterations for both the independence model and the LC model with 2 classes, equal to 134 and 3,926, respectively, with the default tolerance level. However, the computing time is not excessive, being less than 1 second for the first model and 1 minute for the second. Finally, it is also possible to estimate these models using as input the matrix `Y` having dimension 18 and provided by function `freq_data()` as shown above. In this case, however, the computing time is slightly higher.

6 Example based on meningitis data

In order to illustrate the use of the package, we also describe an application about meningitis data collected in an Italian region from 2001 to 2005; we refer to Rossi et al. (2009) for a detailed description and some analyses of the data. Further analyses are reported in Bartolucci and Forcina (2018) and Bartolucci and Forcina (2024) by latent class models for capture-recapture data.

The data are collected in the file 'meningits_data.rda' that we make available through the journal site. The capture occasions are four corresponding to: *HSS* (hospital surveillance of bacterial meningitis); *NDS* (mandatory infectious diseases notifications); *LIS* (the laboratory information system); and *HIS* (hospital information system). There are also some individual covariates: *Age* (binary variable equal to 1 for up to 1 year old and 0 otherwise); *Aez* (binary variable for the recorded type of bacteria, which is equal to 1 for Pneumococcus, Meningococcus, or Tuberculosis and 0 otherwise); *Year* (year of first appearance in a list).

The overall sample size is $n = 944$ with the overall number of captures by list, and also by covariate, shown in Table 1.

	HSS	NDS	LIS	HIS	sample size
overall	355	644	178	826	944
Age=0	261	527	130	689	785
Age=1	94	117	48	137	159
Aez=0	118	290	15	437	516
Aez=1	237	354	163	389	428
Year=2001	46	113	50	149	175
Year=2002	68	112	37	154	175
Year=2003	58	130	34	152	179
Year=2004	66	120	24	182	191
Year=2005	117	169	33	189	224

Table 1: Frequency of captures by list and covariate.

6.1 Analysis without covariates

In the following we show the code to organize the data and estimate the model without covariates from 1 to 3 latent classes, summarizing the main results in an output matrix:

```
# load package
require(LCCR)

# load data
load("meningits_data.rda")
Y = freq_data(D[,1:4])

# estimate with model with 1 to 3 latent classes
est1 = estLCCR(Y,H=1,se_out=TRUE)
est2 = estLCCR(Y,H=2,se_out=TRUE)
est3 = estLCCR(Y,H=3,se_out=TRUE)

# table of results
Tab = rbind(c(k=1,Loglik=est1$lk,np=est1$np,BIC=est1$BIC,N=est1$N),
            c(2,est2$lk,est2$np,est2$BIC,est2$N),
            c(3,est3$lk,est3$np,est3$BIC,est3$N))
```

The output matrix, reporting for each model the value of the maximum log-likelihood, the number of free parameters, the value of BIC, and the estimate of N is as follows:

```
> Tab
     k    Loglik np      BIC         N
[1,] 1 -2948.504  4 5924.409  968.0993
[2,] 2 -2759.193  9 5580.037 1089.0081
[3,] 3 -2754.930 14 5605.762 1234.9012
```

According to the BIC, we select the model with $k = 2$ latent classes. On the other hand, the model with a greater number of classes might not be identifiable. In order to check for the presence of different local maxima, we can also repeat the estimation starting from different parameter values that are randomly generated. For example, with 2 latent classes, we can use the following code:

```
# check starting values
est2r = est2
for(it in 1:5){
  tmp = estLCCR(Y,H=2,init_rand=TRUE)
  if(est2r$lk>est2$lk) est2r = tmp
}
```

According to this analysis, it emerges that the found values of the maximum log-likelihood starting with the deterministic initialization rule are not surpassed by those found starting with the random initialization rule.

In order to obtain a profile confidence interval for the population size, we can simply use command `CI2 = confint(est2)` and then we can show the parameter estimates and the confidence interval by command `summary()` obtaining the output below. The confidence interval can also be represented by command `plot(CI2)`, obtaining the plot in Figure 1. The estimation output is as follows:

```
> summary(est2)

Estimation of latent class models for capture-recapture data

Call:
estLCCR(Y = Y, H = 2, se_out = TRUE)

Available objects:
 [1] "beta"      "lambda"    "lk"        "N"         "np"        "AIC"
 [7] "BIC"       "M"         "tauv"      "phiv"      "Piv"       "Q"
[13] "lk1"       "lk2"       "lk3"       "lk4"       "call"      "Y"
[19] "H"         "model"     "W"         "X"         "biv"       "flag"
[25] "main"      "free_cov"  "free_biv"  "free_flag" "se_out"    "seN"
[31] "sebeta"    "selambda"

   LogLik        np       AIC       BIC
-2759.193     9.000  5536.386  5580.037

Population size:
      est.     s.e.
N 1089.005 32.57444

Parameters affecting the class weights:
                 est.      s.e.    t-test      p-value
class2.int -0.3655014 0.1029695 -3.549609 0.0003858038

Parameters affecting the conditional capture probabilities given the latent class:
                   est.      s.e.     t-test      p-value
class1.main1 -3.7318944 0.4999153  -7.465054 8.326673e-14
class1.main2 -0.7065195 0.1887566  -3.743019 1.818222e-04
class1.main3 -3.1788471 0.2634316 -12.067068 0.000000e+00
class1.main4  0.5757279 0.1988735   2.894946 3.792242e-03
class2.main1  1.1642476 0.2489039   4.677498 2.903962e-06
class2.main2  3.3943767 0.4367170   7.772486 7.771561e-15
class2.main3 -0.6569705 0.1209995  -5.429531 5.650224e-08
class2.main4  2.5736757 0.2194706  11.726744 0.000000e+00

> summary(CI2)

Confidence interval for the population size based on latent class models
```

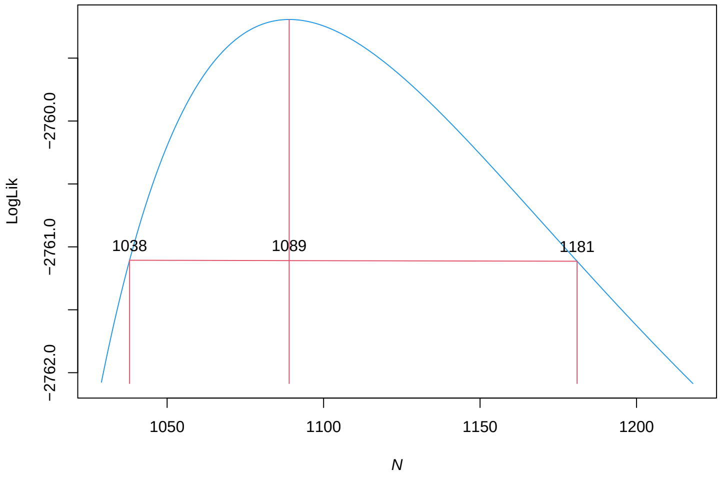

Figure 1: Confidence interval for the latent class model with $k = 2$ classes without covariates.

```
for capture-recapture data

Call:
confint.estLCCR(object = est2, parm = 0.5)

Available objects:
[1] "conf"  "Nv"    "lkv"   "level" "Nh"    "lkh"   "lk1"   "lk2"   "call"

Level:
[1] 0.95

Interval limits:
[1] 1038.005 1181.005
```

Comparing the parameter estimates on the logit scale for the first class (named as `class1.main1` to `class1.main4`) with those for the second class (named as `class2.main1` to `class2.main4`), we note that the latter corresponds to a lower tendency to be captured for all lists. Moreover, considering the intercept of the logit model for the class weights, which is positive, the second class has a higher probability than the first that may be obtained as `exp(est2$be)/(1+exp(est2$be))` and is equal to 0.590. Regarding the estimation of the population size, we have a point estimate of 1,089 with a 95% profile confidence interval equal to (1,037, 1,181).

6.2 Analysis with covariates

In order to illustrate the package for the data with covariates, we consider the model selected for the same data in Bartolucci and Forcina (2018) and also considered in Bartolucci and Forcina (2024). The model, based again on $k = 2$ latent classes, follows a log-linear formulation for the conditional distribution of the capture history given the latent class, with (*i*) main effects depending on the covariate *Year* (centered on 2003) in a way that varies with the latent class and the specific list and (*ii*) bivariate interactions between lists (1,2) and (2,4) that do not depend on the latent class. Moreover, the class weights are affected by the covariates *Age* and *Aez*.

The following commands are used to prepare the data, which are suitably aggregated so as to speed up the routines, and to estimate the model and build the profile confidence interval for the population size.

```
# build covariate matrices
Age = D[,5]
Aez = D[,6]
Year = D[,7]
W = cbind(Age,Aez)
X = Year-2003
agg = aggr_data(Y,W=W,X=X)
Wa = agg$Wa; Xa = cbind(agg$Xa); Ya = agg$Ya
colnames(Xa) = "Year"

# estimate model with covariates
biv = rbind(c(1,2),c(2,4))
est2cov = estLCCR(Ya,model ="loglin",H=2,W=Wa,X=Xa,biv=biv,free_cov="both",
                  free_biv="int",se_out=TRUE)

# estimate model without covariates and aggregated data
est2aggr = estLCCR(Ya,H=2)

# build confidence interval
CI2cov = confint(est2cov,parm=list(step=0.5))
```

In this example, data aggregation is effective because there are $I = 20$ distinct strata despite an overall sample size close to one thousand. Also note that to estimate the model of interest, apart from using arguments W and X, we use argument biv that contains the list of the bivariate interactions, free_cov="both" to require that the effect of the covariate *Year* on the main log-linear parameters varies with list and the latent class, and free_biv="int" to require that the two interactions are distinct. Finally, the model without covariates is estimated on the aggregated data in order to make a fair comparison in terms of log-likelihood and BIC.

To better understand the model structure, for the model with 2 classes it is also possible to obtain the design matrices used to parametrize the conditional distribution of the capture configurations given the covariates and the latent class as follows:

```
M = design_matrix_loglin(J = 4, H = 2, X = Xa, biv = biv, free_cov= "both",
                         free_biv = "int")$M
```

obtaining an array of dimension 16 × 18 × 2, where 16 is the number of capture configurations, 18 is the number of parameters, 2 is the number of classes, and 20 is the number of strata. Note that these design matrices are directly provided among the output arguments of the estimation function.

By the usual command summary(), applied to the previous output objects, it is possible to obtain the main estimation results; by command plot(est2cov) it is also possible to represent the profile log-likelihood function with respect to N, which is reported in Figure 2.

```
> summary(est2cov)

Estimation of latent class models for capture-recapture data

Call:
estLCCR(Y = Ya, H = 2, model = "loglin", W = Wa, X = Xa, biv = biv,
    free_cov = "both", free_biv = "int", se_out = TRUE)

Available objects:
 [1] "beta"      "lambda"    "lk"        "N"         "np"        "AIC"
 [7] "BIC"       "M"         "tauv"      "phiv"      "Piv"       "Q"
[13] "lk1"       "lk2"       "lk3"       "lk4"       "call"      "Y"
```

```
[19] "H"         "model"     "W"         "X"         "biv"       "flag"
[25] "main"      "free_cov"  "free_biv"  "free_flag" "se_out"    "seN"
[31] "sebeta"    "selambda"

   LogLik        np       AIC       BIC
 1298.696    21.000 -2555.392 -2453.540

Population size:
      est.     s.e.
N 1358.617 164.8137

Parameters affecting the class weights:
               est.      s.e.    t-test     p-value
class2.int -3.396401 0.3559948 -9.540592 0.000000000
class2.Age  1.196301 0.4632977  2.582143 0.009818901
class2.Aez  3.881855 0.3764615 10.311426 0.000000000

Parameters affecting the conditional capture probabilities given the latent class:
                        est.       s.e.      t-test      p-value
class1.main1      -3.97671435 0.33631698 -11.8243045 0.000000e+00
class1.main2      -1.88968786 0.37771864  -5.0028981 5.647479e-07
class1.main3      -7.63620950 4.54854883  -1.6788232 9.318650e-02
class1.main4      -0.50990995 0.38807709  -1.3139398 1.888665e-01
class2.main1      -1.70265877 0.32081373  -5.3073127 1.112532e-07
class2.main2      -0.64344325 0.44667633  -1.4405134 1.497222e-01
class2.main3      -0.10490127 0.16023619  -0.6546665 5.126825e-01
class2.main4       1.27829624 0.37543619   3.4048296 6.620536e-04
class1.resp1.Year  0.47673522 0.12227804   3.8987805 9.667837e-05
class1.resp2.Year  0.09851558 0.07392841   1.3325809 1.826694e-01
class1.resp3.Year -1.38492913 2.08737446  -0.6634790 5.070238e-01
class1.resp4.Year  0.17408169 0.09860628   1.7654219 7.749288e-02
class2.resp1.Year  0.31957031 0.10915597   2.9276486 3.415358e-03
class2.resp2.Year  0.02815591 0.15996829   0.1760093 8.602866e-01
class2.resp3.Year -0.16880741 0.08695542  -1.9413098 5.222072e-02
class2.resp4.Year  0.09128870 0.17884609   0.5104316 6.097491e-01
biv1-2             2.82970526 0.29756256   9.5096146 0.000000e+00
biv2-4             1.51790409 0.36899015   4.1136710 3.894163e-05

> summary(CI2cov)

Confidence interval for the population size based on latent class models
for capture-recapture data

Call:
confint.estLCCR(object = est2cov, parm = 0.5)

Available objects:
[1] "conf"  "Nv"    "lkv"   "level" "Nh"    "lkh"   "lk1"   "lk2"   "call"

Level:
[1] 0.95

Interval limits:
[1] 1127.617 1827.117
```

From this output, first of all we note that the BIC of the model with covariates is -2,553.5,

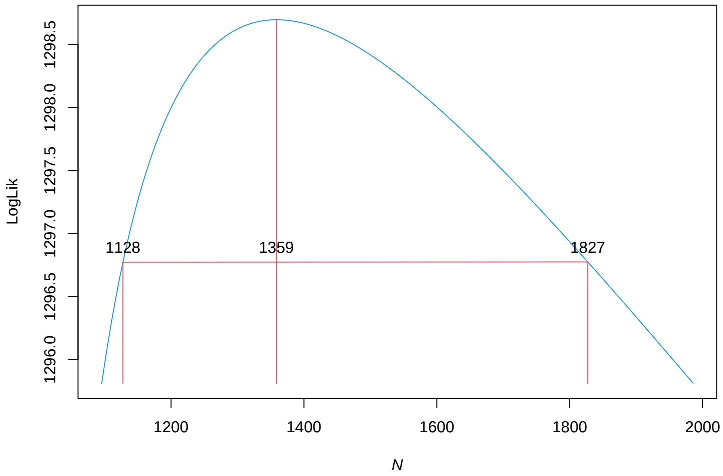

Figure 2: Confidence interval for the latent class model with $k = 2$ classes without covariates.

a value much lower than that of the model without covariates, which is equal to -2,336.7. Moreover, the estimated population size is equal to 1,359 with a 95% confidence interval equal to (1,128, 1,827).

Appendix

Score vector and information matrix

The score vector and the information matrix for $\tilde{\ell}_1(\boldsymbol{\beta})$ are

$$\begin{aligned}\tilde{\boldsymbol{s}}_1(\boldsymbol{\beta}) &= \sum_{i\in\mathcal{I}} \boldsymbol{W}_i'(\tilde{\boldsymbol{n}}_i - \hat{N}_i\boldsymbol{\pi}_i),\\ \tilde{\boldsymbol{F}}_1(\boldsymbol{\beta}) &= \sum_{i\in\mathcal{I}} \hat{N}_i\boldsymbol{W}_i'\boldsymbol{\Omega}(\boldsymbol{\pi}_i)\boldsymbol{W}_i,\end{aligned}$$

where, in general, $\boldsymbol{\Omega}(\boldsymbol{v}) = \text{diag}(\boldsymbol{v}) - \boldsymbol{v}\boldsymbol{v}'$ for any probability vector $\boldsymbol{v}$.

Under the log-linear parametrization, the score vector and information matrix for $\tilde{\ell}_2(\boldsymbol{\beta})$ are

$$\begin{aligned}\tilde{\boldsymbol{s}}_2(\boldsymbol{\lambda}) &= \sum_{i\in\mathcal{I}} \boldsymbol{M}_{h,i}'(\tilde{\boldsymbol{y}}_{h,i} - \tilde{n}_{h,i}\boldsymbol{q}_{h,i}),\\ \tilde{\boldsymbol{F}}_2(\boldsymbol{\lambda}) &= \sum_{i\in\mathcal{I}} \tilde{n}_{h,i}\boldsymbol{M}_{h,i}'\boldsymbol{\Omega}(\boldsymbol{q}_{h,i})\boldsymbol{M}_{h,i}.\end{aligned}$$

Under the recursive logit parametrization, first of all note that on the basis of expression (4) we can reformulate $\tilde{\ell}_2(\boldsymbol{\beta})$ as

$$\tilde{\ell}_2(\boldsymbol{\beta}) = \sum_{h\in\mathcal{H}}\sum_{i\in\mathcal{I}} \tilde{\boldsymbol{y}}_{h,i}'\{\boldsymbol{A}\boldsymbol{M}_{h,i}\boldsymbol{\lambda} - \boldsymbol{B}\log[\boldsymbol{1} + \exp(\boldsymbol{M}_{h,i}\boldsymbol{\lambda})]\}.$$

Consequently, we have the following expression for the score vector and observed information matrix

$$\begin{aligned}\tilde{\boldsymbol{s}}_2(\boldsymbol{\beta}) &= \sum_{h\in\mathcal{H}}\sum_{i\in\mathcal{I}} \boldsymbol{M}_{h,i}'[\boldsymbol{A}' - \text{diag}(\boldsymbol{\rho}_{h,i})\boldsymbol{B}']\tilde{\boldsymbol{y}}_{h,i},\\ \tilde{\boldsymbol{F}}_2(\boldsymbol{\beta}) &= \sum_{h\in\mathcal{H}}\sum_{i\in\mathcal{I}} \boldsymbol{M}_{h,i}'\text{diag}(\boldsymbol{\rho}_{h,i})\text{diag}(\boldsymbol{1} - \boldsymbol{\rho}_{h,i})\text{diag}(\boldsymbol{B}'\tilde{\boldsymbol{y}}_{h,i})\boldsymbol{M}_{h,i},\end{aligned}$$

where

$$\boldsymbol{\rho}_{h,i} = \operatorname{diag}[\mathbf{1} + \exp(\boldsymbol{M}_{h,i}\boldsymbol{\lambda})]^{-1} \exp(\boldsymbol{M}_{h,i}\boldsymbol{\lambda}).$$

Initialization of the EM algorithm

As already mentioned, the EM algorithm can be initialized by a deterministic rule, based on the observed data, or a random rule. These rules consist in choosing the starting values as follows:

- N is initialized as $1.25n$ with the deterministic rule and as un, where $u \sim \text{Unif}(0, 1)$, with the random rule;
- with the deterministic rule, $\boldsymbol{\lambda}$ is initialized as a vector of zeros of suitable dimension when $H = 1$ and by a suitable transformation of the estimates obtained under this model when $H > 1$; with the random rule, the elements of the initial $\boldsymbol{\lambda}$ are drawn from a standard Normal distribution;
- $\boldsymbol{\beta}$ is initialized from a vector of zeros of suitable dimension with the deterministic rule and as a vector of random numbers drawn from a standard Normal distribution with the random rule.

References

H. Akaike. Information theory as an extension of the maximum likelihood principle. In B. N. Petrov and F. Csaki, editors, *Second International Symposium on Information Theory*, pages 267–281, Budapest, 1973. Akademiai Kiado. [p10]

S. Aleshin-Guendel, M. Sadinle, and J. Wakefield. The central role of the identifying assumption in population size estimation. *Biometrics*, 80(1):ujad028, 2024. doi: 10.1093/biomtc/ujad028. [p3]

D. Alunni Fegatelli and L. Tardella. Flexible behavioural capture-recapture modelling. *Biometrics*, 72:125–135, 2016. doi: 10.1111/biom.12417. [p2, 7]

S. Baillargeon and L.-P. Rivest. Rcapture: loglinear models for capture-recapture in R. *Journal of Statistical Software*, 19:1–31, 2007. doi: 10.18637/jss.v019.i05. [p2]

F. Bartolucci and A. Forcina. A class of latent marginal models for capture–recapture data with continuous covariates. *Journal of the American Statistical Association*, 101:786–794, 2006. doi: 10.1198/073500105000000243. [p1]

F. Bartolucci and A. Forcina. Latent class: Rasch models and marginal extensions. In D. Böhning, P. G. M. van der Heijden, and J. Bunge, editors, *Capture-Recapture Methods for the Social and Medical Sciences*, pages 291–304. Chapman and Hall/CRC, 2018. doi: 10.4324/9781315151939-20. [p12, 15]

F. Bartolucci and A. Forcina. Estimating the size of a closed population by modeling latent and observed heterogeneity. *Biometrics*, 80(2):ujae017, 2024. doi: 10.1093/biomtc/ujae017. [p1, 2, 4, 5, 10, 12, 15]

F. Bartolucci and A. Forcina. *LCCR: Latent Class Capture-Recapture Models*, 2025. URL `https://cran.r-project.org/web/packages/LCCR/index.html`. R package version 2.0.1. [p1]

F. Bartolucci and F. Pennoni. A class of latent Markov models for capture–recapture data allowing for time, heterogeneity, and behavior effects. *Biometrics*, 63:568–578, 2007. doi: 10.1111/j.1541-0420.2006.00702.x. [p1]

D. Böhning, J. Bunge, and P. G. M. van der Heijden. *Capture-Recapture Methods for the Social and Medical Sciences*. Chapman and Hall, 2018. doi: 10.4324/9781315151939. [p1]

R. M. Cormack. Interval estimation for mark-recapture studies of closed populations. *Biometrics*, 48:567–576, 1992. doi: 10.2307/2532310. [p2]

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *Journal of the Royal Statistical Society: Series B*, 39: 1–38, 1977. doi: 10.1111/j.2517-6161.1977.tb01600.x. [p5]

Y. Liu, P. Li, and J. Qin. Maximum empirical likelihood estimation for abundance in a closed population from capture-recapture data. *Biometrika*, 104:527–543, 2017. doi: 10.1093/biomet/asx038. [p1, 4, 10]

D. Manrique-Vallier. Bayesian population size estimation using Dirichlet process mixtures. *Biometrics*, 72:1246–1254, 2016. doi: 10.1111/biom.12502. [p2]

D. Manrique-Vallier. *LCMCR: Bayesian Non-Parametric Latent-Class Capture-Recapture*, 2023. URL `https://cran.r-project.org/web/packages/LCMCR/index.html`. R package version 0.4.14. [p2]

D. L. Otis, K. P. Burnham, G. C. White, and D. R. Anderson. Statistical inference from capture data on closed animal populations. *Wildlife Monographs*, 64:3–135, 1978. doi: 10.2307/3830650. [p1, 2]

A. B. Owen. *Empirical Likelihood*. Chapman and Hall/CRC, 2001. doi: 10.1201/9781420036152. [p1]

L.-P. Rivest and S. Baillargeon. Applications and extensions of Chao's moment estimator for the size of a closed population. *Biometrics*, 63:999–1006, 2007. doi: 10.1111/j.1541-0420.2007.00779.x. [p2]

P. G. Rossi, J. Mantovani, E. Ferroni, A. Forcina, E. Stanghellini, F. Curtale, and P. Borgia. Incidence of bacterial meningitis (2001–2005) in Lazio, Italy: the results of an integrated surveillance system. *BMC Infectious Diseases*, 9:1–10, 2009. doi: 10.1186/1471-2334-9-13. [p12]

L. Sanathanan. Estimating the size of a multinomial population. *Annals of Mathematical Statistics*, 43:142–152, 1972. doi: 10.1214/aoms/1177692709. [p1]

G. Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6:461–464, 1978. doi: 10.1214/aos/1176344136. [p10]

B. W. Silverman. Modern slavery: an application of multiple systems estimation. Technical report, Home Office, London, 2014. URL `https://www.gov.uk/government/publications/modern-slavery-an-application-of-multiple-systems-estimation`. [p11]

B. W. Silverman. Multiple-systems analysis for the quantification of modern slavery: classical and Bayesian approaches. *Journal of the Royal Statistical Society, Series A*, 183:691–736, 2020. doi: 10.1111/rssa.12505. [p11]

Francesco Bartolucci
Department of Economics, University of Perugia
Perugia, 06124
Italy
(0000-0001-7057-1421)
`francesco.bartolucci@unipg.it`

Antonio Forcina
Formerly at University of Perugia
Perugia, 06124
Italy
(0000-0001-5239-5495)
`forcinarosara@gmail.com`