Package 'CLCM'

September 22, 2026

Title Estimate Confirmatory Latent Class Models

Version 0.1.1

Description Estimate confirmatory latent class models for a variety of item response types that are encountered in the clinical field. One or two timepoints are supported. Latent regression estimation can be performed, allowing for comparisons of longitudinal latent class assignments (e.g., treatment success/failure) across observed groups (e.g., treatment arms in clinical trials). Fit statistics C2 (a limited-information goodness-of-fit statistic), Akaike Information Criterion (AIC), and Bayesian Information Criterion (BIC) are available as well. Methods are described in Iaconangelo (2026) <doi:10.5281/zenodo.22663151>.

License GPL (>= 3)

URL https://github.com/CJangelo/CLCM, https://cjangelo.github.io/CLCM/

BugReports https://github.com/CJangelo/CLCM/issues

Encoding UTF-8

Imports Matrix, numDeriv, stats, utils

Suggests ggplot2, knitr, nnet, rmarkdown, scales

VignetteBuilder knitr

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Charlie Iaconangelo [aut, cre]

Maintainer Charlie Iaconangelo <charles.iaconangelo@gmail.com>

Repository CRAN

Date/Publication 2026-09-21 22:50:11 UTC

Contents

`aic_bic_clcm` *AIC, BIC, -2LL*

Description

Compute AIC, BIC, and 2LL of the CLCM model

Usage

```
aic_bic_clcm(mod)
```

Arguments

`mod`	object from the `clcm()` function

Value

A list with components `neg_2LL` (-2 times the marginal log-likelihood), `npar` (number of estimated parameters: item parameters plus latent class proportions or latent regression coefficients), `AIC`, and `BIC`. `BIC` uses `N = nrow(mod$dat)`, the number of subject-timepoint rows. Lower values indicate better relative fit.

Examples

```
set.seed(3112021)
# Simulate 5 Ordinal items
sim.dat <- simulate_clcm(N = 200,
                         number.timepoints = 1,
                         item.type = rep('Ordinal', 5),
                         categories.j = rep(4, 5),
                         lc.prop = list('Time_1' = c(0.5, 0.5)) )

# Fit a model with 5 nominal items to the data
mod1 <- clcm(dat = sim.dat$dat,
             item.type = rep('Nominal', 5),
             item.names = sim.dat$item.names,
             Q = sim.dat$Q,
             verbose = FALSE)

# Fit a second model with 5 ordinal items to the data
mod2 <- clcm(dat = sim.dat$dat,
```

```
            item.type = rep('Ordinal', 5),
            item.names = sim.dat$item.names,
            Q = sim.dat$Q,
            verbose = FALSE)

# Compare the two model fits:
unlist(aic_bic_clcm(mod1))
unlist(aic_bic_clcm(mod2))
```

`C2_clcm` *C2 Fit Statistic*

Description

Compute C2 Test Statistic for the CLCM

Usage

```
C2_clcm(mod, verbose = TRUE)
```

Arguments

`mod`	estimated model object from `clcm()` function
`verbose`	logical; print the C2 statistic and associated p-value? Default is TRUE, set to FALSE for simulations.

Value

A list with the following components:

`C2`	the C2 test statistic
`pval`	p-value for the test of exact fit; the null hypothesis is that the model reproduces the first- and second-order marginal probabilities
`rmsea`	root mean square error of approximation, `sqrt(C2 / (N * df))`
`N`	sample size used in the statistic: the number of rows in `mod$dat`
`df`	degrees of freedom: number of marginal probabilities evaluated minus the number of estimated item parameters
`p.obs`	column vector of observed probabilities for each item response pattern
`p.mod`	column vector of model-implied probabilities for the same patterns; plot against `p.obs` to assess fit visually

References

Monroe, S., & Cai, L. (2015). Evaluating structural equation models for categorical outcomes: A new test statistic and a practical challenge of interpretation. Multivariate Behavioral Research, 50(6), 569-583. doi:10.1080/00273171.2015.1032398

Examples

```
set.seed(3112021)
sim.dat <- simulate_clcm(N=200,
                         number.timepoints = 1,
                         item.type = rep('Ordinal', 5),
                         categories.j = rep(2, 5),
                         lc.prop = list('Time_1' = c(0.5, 0.5)) )

mod <- clcm(dat = sim.dat$dat,
            item.type = sim.dat$item.type,
            item.names = sim.dat$item.names,
            Q = sim.dat$Q)

mod.fit <- C2_clcm(mod)
```

`clcm` *Fit CLCM*

Description

Estimate confirmatory latent class model

Usage

```
clcm(
  dat,
  item.type = NULL,
  item.names = NULL,
  Q = NULL,
  lc.con = NULL,
  lat.reg = NULL,
  sv = NULL,
  post.true = NULL,
  initial.lprior = NULL,
  initial.post = NULL,
  max.diff = 1e-04,
  max.it = 1000,
  verbose = TRUE
)
```

Arguments

`dat`	data frame containing item responses. If the data contains more than one time-point, it must be specified by a variable named `Time`, a factor of two levels. Data should be in long format, that is, similar format to that used by the nlme/lme4/glmmTMB R packages

`item.type` character vector specifying the type of item to be modeled. The item type options are as follows:

- `Ordinal` - Ordinal model, 1 slope parameter, C-1 intercept parameters, where C is the number of categories. The slope parameter distinguishes the latent classes.
- `Nominal` - Nominal or Multinomial model, 2*(C-1) parameters. The slope parameters distinguish the latent classes.
- `Poisson` - Poisson model, 2 parameters. The model is parameterized using lambda. See `rpois` for details. The lambda parameter is stratified on the latent classes. The slope parameter distinguishes the latent classes.
- `Neg_Binom` - Negative Binomial model, 3 parameters. The model is parameterized using mu and size. See `rnbinom` for details on parameters. The mu parameter is stratified on the latent classes; the size parameter is common to all latent classes. The slope parameter distinguishes the latent classes.
- `ZINB` - Zero-Inflated Negative Binomial model, 4 parameters. The model is parameterized as mu, size, and zi (zero inflation parameter). The mu parameter is stratified on the latent classes; the size and zi parameters are common to all latent classes. The slope parameter distinguishes the latent classes.
- `ZIP`- Zero-Inflated Poisson model, 3 parameters. The model is parameterized using lambda and zi (zero inflation parameter). The lambda parameter is stratified on latent classes. The zi parameter is common to all latent classes. The slope parameter distinguishes the latent classes.
- `Normal` - Normal distribution model, 2 parameters. The model is parameterized using a mean and variance The mean is stratified on latent classes. The variance is common to all latent classes (pooled across latent classes) The slope parameter distinguishes the latent classes.
- `Beta` - Beta distribution model, 3 parameters. Support ranges from 0 to 1. The model is parameterized using shape1 and shape2 parameters. See `rbeta` for details The shape1 parameter is stratified on latent classes The shape 2 parameter is common to all latent classes The slope parameter distinguishes the latent classes.

`item.names` specify the item names; the dataframe column names containing item responses

`Q` optional pass the Q-matrix, default is K=1, two latent classes. This is a confirmatory latent class model, hence the need for the specification. Note that only dichotomous attributes (factors) are supported. Only the conjunctive condensation rule is supported. All condensation rules are equivalent for items that evaluate a single attribute (factor).

`lc.con` optional list of constraints on each latent class at each timepoint, where `NA` constrains that latent class to zero at that timepoint. For example, constraint latent class one (out of two) to be zero at timepoint 1: `lc.con = list('Time_1' = c(NA, 1), 'Time_2' = c(1, 1))`. Note that the `dat` dataframe passed must contain the variable `Time`.

`lat.reg` optional pass variables to regress latent class on For example, regress the latent classes at timepoint 2 onto the Group variable: `lat.reg = list('Time_1' = NULL, 'Time_2' = 'Group' )`, where `Time_1` and `Time_2` are the discrete timepoints. Note that the `dat` dataframe passed must contain the variable `Time`.

sv	optional list of the starting values for item parameter estimation
post.true	optional matrix of the true posterior distributions, as produced by simulate_clcm(); used to evaluate parameter recovery in simulation studies
initial.lprior	optional matrix of the initial log-prior distribution. This is important for the 2-stage estimation routine.
initial.post	optional matrix of the initial posterior distribution
max.diff	convergence tolerance of item param estimation; default is 1e-04
max.it	maximum number of iterations in EM estimation procedure; default is 1e3
verbose	logical; should the function print the estimation progress? Default is TRUE, recommend set to FALSE for simulations.

Value

A list with the following components:

item.param	list of length J (number of items); each element holds the estimated parameters for one item, named as described under item.type
lat.reg.param	matrix of latent regression coefficients (design-matrix columns by 2^K - 1 latent classes), or NULL if no latent regression was specified in lat.reg
dat	the input data frame with appended columns: lprior_LC_* (log prior probability of each latent class), post_LC_* (posterior probability of each latent class), and lca (modal latent class assignment, an integer in 1:2^K)
item.type, item.names, Q, lc.con, lat.reg	the corresponding inputs, as used in estimation (with defaults filled in where they were NULL)
categories.j	list of length J; for categorical and count items, the vector of observed categories; NA for Normal and Beta items
lprior.names, post.names	character vectors naming the log-prior and posterior columns in dat
K	number of attributes (factors); the model has 2^K latent classes
alpha	2^K by K matrix of attribute patterns, one row per latent class
eta	2^K by J matrix from the conjunctive condensation rule; entry (l, j) is 1 if latent class l possesses every attribute item j measures

References

Iaconangelo, C. (2026). Confirmatory Restricted Latent Class Models for Categorical, Count, and Continuous Response Types via the EM Algorithm. doi:10.5281/zenodo.22663151

Examples

```
set.seed(3112021)
sim.dat <- simulate_clcm(N = 200,
                         number.timepoints = 1,
                         item.type = rep('Ordinal', 5),
                         categories.j = rep(4, 5),
```

```
                        lc.prop = list('Time_1' = c(0.5, 0.5)) )

mod <- clcm(dat = sim.dat$dat,
            item.type = sim.dat$item.type,
            item.names = sim.dat$item.names,
            Q = sim.dat$Q,
            verbose = FALSE)
```

`clcm_2stage` *Two-Stage Estimation of the CLCM*

Description

Treat item parameters as fixed to fit a longitudinal CLCM. The model can have any number of timepoints. The only limitation is the estimation of the posteriors - item parameters are fixed. The CLCM R package requires that count items start at 0.

Usage

```
clcm_2stage(dat, clcm.mod, lprior.t1)
```

Arguments

`dat`	dataframe containing the new item responses you want to fit the model to.
`clcm.mod`	a CLCM model that you already fit (Stage 1). This will have the item parameter estimates, item names, item type, and Q-matrix.
`lprior.t1`	the log-prior used to initialize the first timepoint in `dat`; an N by L matrix, where L is the number of latent classes. Use `compute_lprior()` on the Stage 1 posteriors to obtain this.

Value

Returns a dataframe containing the item responses in `dat` together with the posterior probabilities of latent class membership computed at each timepoint under the fixed Stage 1 item parameters.

References

Iaconangelo, C. (2026). Confirmatory Restricted Latent Class Models for Categorical, Count, and Continuous Response Types via the EM Algorithm. doi:10.5281/zenodo.22663151

Bakk, Z., Tekle, F. B., & Vermunt, J. K. (2013). Estimating the association between latent class membership and external variables using bias-adjusted three-step approaches. Sociological Methodology, 43(1), 272-311. doi:10.1177/0081175012470644

Vermunt, J. K. (2010). Latent class modeling with covariates: Two improved three-step approaches. Political Analysis, 18(4), 450-469. doi:10.1093/pan/mpu003

Examples

```
set.seed(3112021)
sim.dat <- simulate_clcm(N = 200,
                         number.timepoints = 2,
                         item.type = rep('Ordinal', 5),
                         categories.j = rep(4, 5),
                         lc.prop = list('Time_1' = c(0.5, 0.5),
                                        'Time_2' = c(0.5, 0.5)) )

# Stage 1: fit the model to the first timepoint only
dat.t1 <- sim.dat$dat[sim.dat$dat$Time == 'Time_1', ]
mod1 <- clcm(dat = dat.t1,
             item.type = sim.dat$item.type,
             item.names = sim.dat$item.names,
             Q = sim.dat$Q,
             verbose = FALSE)

# Stage 2: fixed item parameters, estimate posteriors at timepoint 2
lprior1 <- compute_lprior(post = mod1$dat[, mod1$post.names],
                          type = 'standard_EB',
                          K = mod1$K)
dat.t2 <- sim.dat$dat[sim.dat$dat$Time == 'Time_2', ]
post2 <- clcm_2stage(dat = dat.t2, clcm.mod = mod1, lprior.t1 = lprior1)
head(post2)
```

`compute_lprior` *Prior Distribution*

Description

Compute the log of the prior distribution for use in the EM algorithm

Usage

```
compute_lprior(post, K, Z = NULL, type = "standard_EB", reg.formula = NULL)
```

Arguments

`post`	posterior distribution
`K`	integer, the number of attributes (factors); the model has 2^K latent classes
`Z`	data frame or matrix containing covariates
`type`	the type of prior distribution - this is selected automatically in the clcm() function depending on what is passed. Default is 'standard_EB', the other option is 'latent_regression'.
`reg.formula`	if a latent regression is estimated, pass the model specification as a character here

Value

For `type = 'standard_EB'`, an N by 2^K matrix of log prior probabilities (each row identical, the log of the mean posterior). For `type = 'latent_regression'`, a list with `vP` (matrix of regression coefficients, design-matrix columns by 2^K - 1) and `lprior` (the N by 2^K matrix of log prior probabilities implied by the regression).

References

Iaconangelo, C. (2026). Confirmatory Restricted Latent Class Models for Categorical, Count, and Continuous Response Types via the EM Algorithm. doi:10.5281/zenodo.22663151

Wayman, E. A., Culpepper, S. A., Douglas, J., & Bowers, J. (2025). A restricted latent class model with polytomous attributes and respondent-level covariates. Behaviormetrika, 1-29. doi:10.1007/s41237025002718

Wayman, E. A., Culpepper, S. A., Douglas, J., & Bowers, J. (2025). A restricted latent class hidden Markov model for polytomous responses, polytomous attributes, and covariates: Identifiability and application. Journal of Educational and Behavioral Statistics. doi:10.3102/10769986251415569

Examples

```
set.seed(3112021)
sim.dat <- simulate_clcm(N = 200,
                         number.timepoints = 1,
                         item.type = rep('Ordinal', 5),
                         categories.j = rep(4, 5),
                         lc.prop = list('Time_1' = c(0.5, 0.5)) )
mod <- clcm(dat = sim.dat$dat,
            item.type = sim.dat$item.type,
            item.names = sim.dat$item.names,
            Q = sim.dat$Q,
            verbose = FALSE)
lprior <- compute_lprior(post = mod$dat[, mod$post.names],
                         type = 'standard_EB',
                         K = mod$K)
head(exp(lprior))
```

`simulate_clcm` *Simulate Data*

Description

Simulate data for the CLCM

Usage

```
simulate_clcm(
  N,
  number.timepoints,
  Q = NULL,
  item.type = NULL,
  categories.j = NULL,
  transition.matrix = NULL,
  lc.prop = NULL,
  post = NULL,
  param = NULL,
  item.names = NULL
)
```

Arguments

N	integer specifying the sample size
number.timepoints	integer specify the number of timepoints, 1 or 2
Q	the Q-matrix, a matrix of 1s and 0s specifying the factor loading structure. The default is 1 factor (K=1), which forms two latent classes
item.type	character vector specifying the type of item to be modeled

- `Ordinal` - Ordinal model, 1 slope parameter, C-1 intercept parameters, where C is the number of categories. The slope parameter distinguishes the latent classes.
- `Nominal` - Nominal or Multinomial model, 2*(C-1) parameters. The slope parameters distinguish the latent classes.
- `Poisson` - Poisson model, 2 parameters. The model is paramterized using lambda. See ?rpois for details. The lambda parameter is stratified on the latent classes. The slope parameter distinguishes the latent classes.
- `Neg_Binom` - Negative Binomial model, 3 parameters. The model is parameterized using mu and size. See ?rnbinom for details on parameters. The mu parameter is stratified on the latent classes; the size parameter is common to all latent classes. The slope parameter distinguishes the latent classes.
- `ZINB` - Zero-Inflated Negative Binomial model, 4 parameters. The model is parameterized as mu, size, and zi (zero inflation parameter). The mu parameter is stratified on the latent classes; the size and zi parameters are common to all latent classes. The slope parameter distinguishes the latent classes.
- `ZIP`- Zero-Inflated Poisson model, 3 parameters. The model is parameterized using lambda and zi (zero inflation parameter). The lambda parameter is stratified on latent classes. The zi parameter is common to all latent classes. The slope parameter distinguishes the latent classes.
- `Normal` - Normal distribution model, 2 parameters. The model is parameterized using a mean and variance The mean is stratified on latent classes.

The variance is common to all latent classes (pooled across latent classes) The slope parameter distinguishes the latent classes.

- `Beta` - Beta distribution model, 3 parameters. Support ranges from 0 to 1. The model is parameterized using shape1 and shape2 parameters. See ?rbeta for details The shape1 parameter is stratified on latent classes The shape 2 parameter is common to all latent classes The slope parameter distinguishes the latent classes.

`categories.j` numeric vector specifying the number of categories of each item. For 'Normal' or 'Beta' item types, the value should be NA

`transition.matrix`
a 2^K by 2^K numeric matrix that specifies the transition probabilities. This is used in conjunction with the lc.prop at timepoint 1. See Vignettes for a detailed example.

`lc.prop` list of the latent class proportions at each timepoint. For example, `lc.prop = list('Time_1' = c(0, 1), 'Time_2' = c(0.6, 0.4))` specifies that at timepoint 1 all subjects are in latent class 2, and at timepoint 2, 40% are in latent class 2, with the remainder in latent class 1

`post` a matrix of the true posterior distributions - long data format. Generate the posterior distributions according to user-preference, then pass posterior distributions to the function. This is the preferred way to specify the latent class proportions and transition probabilities because it offers maximum control and flexibility. See Vignettes for detailed examples on generating posterior distributions.

`param` list of item parameters, default is to use the values in the function

`item.names` character vector of item names

Value

A list with the following components:

`dat` data frame in long format with one row per subject per timepoint: `USUBJID`, `Time`, one column per item, `true_post_LC_*` (generating latent class membership, one-hot), and `true_lca` (generating latent class, an integer in `1:2^K`)

`item.responses` numeric matrix of the item responses only (rows match `dat`, columns are `item.names`)

`post` numeric matrix of generating latent class membership (rows match `dat`, 2^K columns, one-hot)

`lca` integer vector of generating latent class per row of `dat`

`param` list of generating item parameters, one element per item

`item.type`, `item.names`, `categories`, `lc.prop`, `Q`
the corresponding inputs, with defaults filled in where they were `NULL`

`K`, `alpha`, `eta` as in the return value of `clcm()`

References

Iaconangelo, C. (2026). Confirmatory Restricted Latent Class Models for Categorical, Count, and Continuous Response Types via the EM Algorithm. doi:10.5281/zenodo.22663151

Wayman, E. A., Culpepper, S. A., Douglas, J., & Bowers, J. (2025). A restricted latent class model with polytomous attributes and respondent-level covariates. Behaviormetrika, 1-29. doi:10.1007/s41237025002718

Wayman, E. A., Culpepper, S. A., Douglas, J., & Bowers, J. (2025). A restricted latent class hidden Markov model for polytomous responses, polytomous attributes, and covariates: Identifiability and application. Journal of Educational and Behavioral Statistics. doi:10.3102/10769986251415569

Culpepper, S. A. (2019). An exploratory diagnostic model for ordinal responses with binary attributes: Identifiability and estimation. Psychometrika, 84(4), 921-940. doi:10.1007/s11336019-096834

Liu, Y., & Culpepper, S. A. (2024). Restricted latent class models for nominal response data: Identifiability and estimation. Psychometrika, 89(2), 592-625. doi:10.1007/s11336023099407

Minchen, N. D., de la Torre, J., & Liu, Y. (2017). A cognitive diagnosis model for continuous response. Journal of Educational and Behavioral Statistics, 42(6), 651-677. doi:10.3102/1076998617703060

Examples

```
set.seed(3112021)
sim.dat <- simulate_clcm(N = 50, number.timepoints = 1,
               item.type = rep('Ordinal', 5),
               categories.j = rep(4, 5),
               lc.prop = list('Time_1' = c(0.5, 0.5)) )
str(sim.dat$dat)
```

---

start_with_zero *Count Items Start with Zero*

Description

The CLCM R package requires that count items start at 0. This helper shifts each specified item so that its minimum observed value is 0.

Usage

```
start_with_zero(dat, items)
```

Arguments

`dat`	dataframe containing the item responses
`items`	vector of item names. These should be the count items that need to start at zero in order for `clcm` function to estimate the model.

Value

Returns the dataframe `dat` with the columns named in `items` shifted so that the minimum observed value of each is 0.

Examples

```
dat <- data.frame(Item_1 = c(3, 5, 4, 7),
                  Item_2 = c(10, 12, 11, 15))
start_with_zero(dat, items = c('Item_1', 'Item_2'))
```

`transition_matrix_clcm` *Transition Matrix*

Description

Compute Transition Matrix, stratified on categorical variable Key to evaluating group differences

Usage

```
transition_matrix_clcm(
  mod,
  eap.classification = FALSE,
  threshold = 0.5,
  modal.classification = FALSE,
  stratification = FALSE,
  covariate = NULL
)
```

Arguments

`mod`	estimated model object from clcm() function. Note that if estimating the transition matrix stratified on a covariate, that (categorical) covariate must be part of the dataframe (dat) that was used to estimate the model, i.e., mod$dat must contain the covariate
`eap.classification`	logical; select if expected a posteriori (EAP) classification is desired If neither MAP nor EAP classification is selected, then sample-level averages will be computed for each latent class. That is, the probabilistic classifications in the subject posterior distributions will be retained and averaged.
`threshold`	numeric value, if EAP classification is selected, must choose a threshold for classification as 1 versus 0 on each attribute (factor).
`modal.classification`	logical; classify subjects using maximum a posteriori (MAP) classification?
`stratification`	logical; should the transition matrix be computed stratified on categorical covariate?
`covariate`	categorical variable, separate transition matrix estimated for each level of the variable. Note that if estimating the transition matrix stratified on a covariate, that (categorical) covariate must be part of the dataframe (dat) that was used to estimate the model, i.e., mod$dat must contain the covariate.

Value

Returns a 2^K by 2^k numeric matrix; if transition matrix is stratified on covariate, then returns a list of 2^K by 2^K numeric matrices.

References

Iaconangelo, C. (2026). Confirmatory Restricted Latent Class Models for Categorical, Count, and Continuous Response Types via the EM Algorithm. doi:10.5281/zenodo.22663151

Bartolucci, F., Farcomeni, A., & Pennoni, F. (2010). An overview of latent Markov models for longitudinal categorical data. arXiv:1003.2804.

Ip, E., Zhang, Q., Rejeski, J., Harris, T., & Kritchevsky, S. (2013). Partially ordered mixed hidden Markov model for the disablement process of older adults. Journal of the American Statistical Association, 108(502), 370-384. doi:10.1080/01621459.2013.770307

Ip, E. H., Zhang, Q., Schwartz, R., Tooze, J., Leng, X., Han, H., & Williamson, D. A. (2013). Multi-profile hidden Markov model for mood, dietary intake, and physical activity in an intervention study of childhood obesity. Statistics in Medicine, 32(19), 3314-3331. doi:10.1002/sim.5719

Wayman, E. A., Culpepper, S. A., Douglas, J., & Bowers, J. (2025). A restricted latent class model with polytomous attributes and respondent-level covariates. Behaviormetrika, 1-29. doi:10.1007/s41237025002718

Wayman, E. A., Culpepper, S. A., Douglas, J., & Bowers, J. (2025). A restricted latent class hidden Markov model for polytomous responses, polytomous attributes, and covariates: Identifiability and application. Journal of Educational and Behavioral Statistics. doi:10.3102/10769986251415569

Examples

```
set.seed(3112021)
sim.dat <- simulate_clcm(N = 200,
                         number.timepoints = 2,
                         item.type = rep('Ordinal', 5),
                         categories.j = rep(4, 5),
                         lc.prop = list('Time_1' = c(0.5, 0.5),
                                        'Time_2' = c(0.5, 0.5)) )

mod <- clcm(dat = sim.dat$dat,
            item.type = sim.dat$item.type,
            item.names = sim.dat$item.names,
            Q = sim.dat$Q,
            verbose = FALSE)

tau.hat <- transition_matrix_clcm(mod)
tau.hat
```

Index