Package: ordinalClust (via r-universe)

May 21, 2026

Type Package

Title Ordinal Data Clustering, Co-Clustering and Classification

Version 1.3.5.1

Date 2026-04-17

Maintainer Julien Jacques <julien.jacques@univ-lyon2.fr>

Description Ordinal data classification, clustering and co-clustering using model-based approach with the BOS (Binary Ordinal Search) distribution for ordinal data (Christophe Biernacki and Julien Jacques (2016) <doi:10.1007/s11222-015-9585-2>).

License GPL (>= 2)

Imports Rcpp (>= 0.12.11), methods

LinkingTo Rcpp, RcppProgress, RcppArmadillo, BH

Suggests knitr, rmarkdown, caret, ggplot2

VignetteBuilder knitr

LazyData true

Depends R (>= 3.3)

NeedsCompilation yes

Author Margot Selosse [aut], Julien Jacques [aut, cre], Christophe Biernacki [aut]

Repository https://cran.r-universe.dev

Date/Publication 2026-04-21 22:03:59 UTC

RemoteUrl https://github.com/cran/ordinalClust

RemoteRef HEAD

RemoteSha 989959cc6b5a33c3bfeb6fb073cd35b5acff4b62

Contents

`bosclassif` *Function to perform a classification*

Description

This function performs a classification algorithm on a dataset with ordinal features, and a label variable that belongs to (1,2,...,kr). The classification function provides two classification models. The first model, (chosen by the argument kc=0), is a multivariate BOS model with the assumtion that, conditional on the class of the observations, the features are independent. The second model is a parsimonious version of the first model. Parsimony is introduced by grouping the features into clusters (as in co-clustering) and assuming that the features of a cluster have a common distribution.

Usage

```
bosclassif(x, y, idx_list=c(1), kr, kc=0, init, nbSEM, nbSEMburn,
          nbindmini, m=0, percentRandomB=0)
```

Arguments

`x`	Matrix made of ordinal data of dimension N*Jtot. The features with same numbers of levels must be placed side by side. The missing values should be coded as NA.
`y`	Vector of length N. It should represent the classes corresponding to each row of x. Must be labeled with numbers (1,2,...,kr).
`idx_list`	Vector of length D. This argument is useful when variables have different numbers of levels. Element d should indicate where the variables with number of levels m[d] begin in matrix x.
`kr`	Number of row classes.
`kc`	Vector of length D. The d-th element indicates the number of column clusters. Set to 0 to choose a classical multivariate BOS model.
`m`	Vector of length D. The d-th element defines the number of levels of the ordinal data.
`nbSEM`	Number of SEM-Gibbs iterations realized to estimate parameters.
`nbSEMburn`	Number of SEM-Gibbs burn-in iterations for estimating parameters. This parameter must be inferior to nbSEM.

nbindmini	Minimum number of cells belonging to a block.
init	String that indicates the kind of initialisation. Must be one of the following strings: "kmeans", "random" or "randomBurnin".
percentRandomB	Vector of length 1. Indicates the percentage of resampling when init is equal to "randomBurnin".

Value

Return an object. The slots are:

@zr	Vector of length N with resulting row partitions.
@zc	List of length D. The d-th item is a vector of length J[d] representing the column partitions for the group of variables d.
@J	Vector of length D. The d-th item represents the number of columns for d-th group of variables.
@W	List of length D. Item d is a matrix of dimension J*kc[d] such that W[j,h]=1 if j belongs to cluster h.
@V	Matrix of dimension N*kr such that V[i,g]=1 if i belongs to cluster g.
@icl	ICL value for co-clustering.
@kr	Number of row classes.
@name	Name of the result.
@number_distrib	Number of groups of variables.
@pi	Vector of length kr. Row mixing proportions.
@rho	List of length D. The d-th item represents the column mixing proportion for the d-th group of variables.
@dlist	List of length d. The d-th item represents the indexes of group of variables d.
@kc	Vector of length D. The d-th element represents the number of clusters column H for the d-th group of variables.
@m	Vector of length D. The d-th element represents the number of levels of the d-th group of variables.
@nbSEM	Number of SEM-Gibbs algorithm iteration.
@params	List of length D. The d-th item represents the blocks parameters for a group of variables d.
@xhat	List of length D. The d-th item represents the dataset of the d-th group of variables, with missing values completed.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
# loading the real dataset
data("dataqol.classif")

set.seed(5)

# loading the ordinal data
M <- as.matrix(dataqol.classif[,2:29])


# creating the classes values
y <- as.vector(dataqol.classif$death)


# sampling datasets for training and to predict
nb.sample <- ceiling(nrow(M)*2/3)
sample.train <- sample(1:nrow(M), nb.sample, replace=FALSE)

M.train <- M[sample.train,]
M.validation <- M[-sample.train,]
nb.missing.validation <- length(which(M.validation==0))
m <- c(4)
M.validation[which(M.validation==0)] <- sample(1:m, nb.missing.validation,replace=TRUE)


y.train <- y[sample.train]
y.validation <- y[-sample.train]



# configuration for SEM algorithm
nbSEM=50
nbSEMburn=40
nbindmini=1
init="kmeans"

# number of classes to predict
kr <- 2
# different kc to test with cross-validation
kcol <- 1


res <- bosclassif(x=M.train,y=y.train,kr=kr,kc=kcol,m=m,
                  nbSEM=nbSEM,nbSEMburn=nbSEMburn,
                  nbindmini=nbindmini,init=init)

predictions <- predict(res, M.validation)
```

`bosclust` *Function to perform a clustering*

Description

This function performs a clustering algorithm on ordinal data by using the multiple latent block model (see references for further details). It allows the user to define D groups of variables that have different numbers of levels. The BOS distribution is used, and the parameters inference is obtained using the SEM-Gibbs algorithm.

Usage

```
bosclust(x, idx_list=c(1), kr, init, nbSEM, nbSEMburn,
        nbindmini, m=0, percentRandomB=0)
```

Arguments

`x`	Matrix made of ordinal data of dimension N*Jtot. The features with the same numbers of levels must be placed side by side. The missing values should be coded as NA.
`idx_list`	Vector of length D. This argument is useful when variables have different numbers of levels. Element d should indicate where the variables with number of levels m[d] begin in matrix x.
`kr`	Number of row clusters.
`m`	Vector of length D. The d-th element defines the number of levels of the ordinal data.
`nbSEM`	Number of SEM-Gibbs iterations realized to estimate the parameters.
`nbSEMburn`	Number of SEM-Gibbs burn-in iterations for estimating parameters. This parameter must be inferior to nbSEM.
`nbindmini`	Minimum number of cells belonging to a block.
`init`	String that indicates the kind of initialisation. Must be one of the following words : "kmeans", "random" or "randomBurnin".
`percentRandomB`	Vector of length 1. Indicates the percentage of resampling when init is equal to "randomBurnin".

Value

`@V`	Matrix of dimension N*kr such that V[i,g]=1 if i belongs to cluster g.
`@zr`	Vector of length N with resulting row partitions.
`@pi`	Vector of length kr. This corresponds to the row mixing proportions.
`@m`	Vector of length D. The d-th element represents the number of levels of the d-th group of variables.
`@icl`	ICL value for clustering.

@name	Name of the result.
@params	List of length D. The d-th item stores the resulting position and precision parameters mu and pi.
@paramschain	List of length nbSEMburn. The parameters of the blocks are stored for each iteration of the SEM-Gibbs algorithm.
@xhat	List of length D. The d-th item represents the dataset of the d-th group of variables, with missing values completed.
@zrchain	Matrix of dimension nbSEM*N. Row i represents the row cluster partitions at iteration i.
@pichain	List of length nbSEM. Item i is a vector of length kr that contains the row mixing proportions at iteration i.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
library(ordinalClust)
data("dataqol")
set.seed(5)

# loading the ordinal data
M <- as.matrix(dataqol[,2:29])

m = 4

krow = 4

nbSEM=50
nbSEMburn=40
nbindmini=2
init = "random"


object <- bosclust(x=M,kr=krow, m=m, nbSEM=nbSEM,
    nbSEMburn=nbSEMburn, nbindmini=nbindmini, init=init)
```

`boscoclust` *Function to perform a co-clustering*

Description

This function runs a co-clustering algorithm on ordinal data by using the latent block model (see references for further details). A BOS distribution is used, and the parameters inference is obtained using the SEM-Gbbs algorithm.

Usage

```
boscoclust(x=matrix(0,nrow=1,ncol=1), idx_list=c(1), kr, kc, init, nbSEM, nbSEMburn,
          nbRepeat=1, nbindmini, m=0, percentRandomB=0)
```

Arguments

x	Matrix made of ordinal data of dimension N*Jtot. The features with the same numbers of levels must be placed side by side. The missing values should be coded as NA.
idx_list	Vector of length D. This argument is useful when variables have different numbers of levels. Element d should indicate where the variables with number of levels m[d] begin in matrix x.
kr	Number of row classes.
kc	Vector of length D. The d-th element indicates the number of column clusters.
m	Vector of length D. The d-th element defines the number of levels of the ordinal data.
nbSEM	Number of SEM-Gibbs iterations realized to estimate parameters.
nbSEMburn	Number of SEM-Gibbs burn-in iterations for estimating parameters. This parameter must be inferior to nbSEM.
nbRepeat	Number of times sampling on rows and columns will be done for each SEM-Gibbs iteration.
nbindmini	Minimum number of cells belonging to a block.
init	String that indicates the kind of initialisation. Must be one of the following words : "kmeans", "random" or "randomBurnin".
percentRandomB	Vector of length 2. Indicates the percentage of resampling when init is equal to "randomBurnin".

Value

@V	Matrix of dimension N*kr such that V[i,g]=1 if i belongs to cluster g.
@icl	ICL value for co-clustering.
@name	Name of the result.
@paramschain	List of length nbSEMburn. The parameters of the blocks are stored for each iteration of the SEM-Gibbs algorithm.
@pichain	List of length nbSEM. Item i is a vector of length kr that contains the row mixing proportions at iteration i.
@rhochain	List of length nbSEM. Item i is a list of length D whose d-th element contains the column mixing proportions of the group of variables d, for iteration i.
@zc	List of length D. The d-th item is a vector of length J[d] representing the column partitions for the group of variables d.
@zr	Vector of length N with resulting row partitions.
@W	List of length D. Item d is a matrix of dimension J*kc[d] such that W[j,h]=1 if j belongs to cluster h.

@m	Vector of length D. The d-th element represents the number of levels of d-th group of variables.
@params	List of length D. The d-th item represents the blocks parameters for a group of variables d.
@pi	Vector of length kr. This corresponds to the row mixing proportions.
@rho	List of length D. The d-th item represents the column mixing proportion for the d-th group of variables.
@xhat	List of length D. The d-th item represents the dataset of the d-th group of variables, with missing values completed.
@zrchain	Matrix of dimension nbSEM*N. Row i represents the row cluster partitions at iteration i.
@zrchain	List of length D. Item d is a matrix of dimension nbSEM*J[d]. Row i represents the column cluster partitions at iteration i.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
library(ordinalClust)

# loading the real dataset
data("dataqol")
set.seed(5)

# loading the ordinal data
M <- as.matrix(dataqol[,2:29])


# defining different number of categories:
m=4


# defining number of row and column clusters
krow = 4
kcol = 4

# configuration for the inference
nbSEM=50
nbSEMburn=40
nbindmini=2
init = "randomBurnin"
percentRandomB=c(20,20)



# Co-clustering execution
```

```
object <- boscoclust(x=M,kr=krow,kc=kcol,m=m,nbSEM=nbSEM,
      nbSEMburn=nbSEMburn, nbindmini=nbindmini, init=init, percentRandomB=percentRandomB)
```

`dataqol` *Questionnaire Responses Of Patients Affected By Breast Cancer*

Description

This dataset contains the responses of 121 patients to 30 questions about their quality of life.

Usage

```
dataqol
```

Format

A dataframe with 121 lines and 31 columns. A line represents a patient and a column contains information about the patients.

Id patient Id

q1-q28 responses to 28 questions with the number of levels equal to 4

q29-q30 responses to 22 questions with the number of levels equal to 7

Source

The table was determined based on data associated with the package available on: https://cran.r-project.org/package=QoLR

`dataqol.classif` *Questionnaire Responses Of Patients Affected By Breast Cancer*

Description

This dataset contains the responses of 40 patients to 30 questions about their quality of life. Furthermore, a variable indicates if the patient survived the disease.

Usage

```
dataqol.classif
```

Format

A dataframe with 40 lines and 32 columns. A line represents a patient and a column contains information about the patients.

Id patient Id

q1-q28 responses to 28 questions with the number of levels equal to 4

q29-q30 responses to 22 questions with the number of levels equal to 7

death if the patient survived (1) or not (2)

Source

The table was determined based on data associated with the package available at: https://cran.r-project.org/package=QoLR

`Msimulated` *Matrix of simulated ordinal data*

Description

This is a toy dataset for running simple examples.

Usage

```
Msimulated
```

Format

An ordinal data matrix with 60 lines and 50 columns. The number of levels is equal to 3. Four blocks are simulated with (mu,pi) parameters equal to (3,0.5), (2,0.7), (1,0.8) and (2,0.6).

`pejSim` *pejSim*

Description

This function computes the probabilty for a level ej to be sampled from a BOS distribution of parameters (mu,pi), with the number of levels equal to m. It can be used to generate data with a BOS distribution.

Usage

```
pejSim(ej,m,mu,p)
```

Arguments

ej	Levels to be sampled
m	Number of levels.
mu	mu parameter for BOS distribution.
p	pi parameter for BOS distribution.

Value

Returns the probability of ej to be sampled from a BOS distribution of parameters (mu,pi), with the number of levels equal to m.

Author(s)

Margot Selosse, Julien Jacques, Christophe Biernacki.

Examples

```
library(ordinalClust)
data("dataqol")
set.seed(5)

m=7
nr=10000
mu=5
pi=0.5

probaBOS=rep(0,m)
for (im in 1:m) probaBOS[im]=pejSim(im,m,mu,pi)
M <- sample(1:m,nr,prob = probaBOS, replace=TRUE)
```

---

plot *~~ Method for the function* `plot` *in the* **ordinalClust** *package ~~*

Description

Plots the result of a classification, clustering or co-clustering performed using the following functions: bosclassif,bosclust,boscoclust.

Methods

`signature(object = "ResultClassifOrdinal")`

`signature(object = "ResultClustOrdinal")`

`signature(object = "ResultCoclustOrdinal")`

`predict` ~~ *Method for the function* `predict` *in the* **ordinalClust** *package* ~~

Description

Method for the function `predict` in the **ordinalClust** package.

Methods

`signature(object = "ResultClassifOrdinal")` Use this method with the result of the function bosclassif, and a new sample to predict the classes.

`summary` ~~ *Methods for the Function* `summary` *in the* **ordinalClust** *package* ~~

Description

Prints the result of a classification, clustering or co-clustering performed using the following functions: bosclassif,bosclust,boscoclust.

Methods

`signature(object = "ResultClassifOrdinal")`

`signature(object = "ResultClustOrdinal")`

`signature(object = "ResultCoclustOrdinal")`

Index