Package ‘BNPMIXcluster’

November 30, 2020

**Type** Package

**Title** Bayesian Nonparametric Model for Clustering with Mixed Scale Variables

**Version** 1.3

**Date** 2020-11-28

**Description** Model-based approach for clustering of multivariate data, capable of combining different types of variables (continuous, ordinal and nominal) and accommodating for different sampling probabilities in a complex survey design. The model is based on a location mixture model with a Poisson-Dirichlet process prior on the location parameters of the associated latent variables. Details of the underlying model is described in Carmona, C., Nieto-Barajas, L. E., Canale, A. (2016) <arXiv:1612.00083>.

**License** MIT + file LICENSE

**URL** https://github.com/christianu7/BNPMIXcluster

**BugReports** https://github.com/christianu7/BNPMIXcluster/issues

**LazyData** TRUE

**Depends** R (>= 2.10),

**Imports** compiler, gplots, MASS, matrixcalc, mvtnorm, plyr, Rcpp (>= 1.0.5), truncnorm

**LinkingTo** Rcpp, RcppArmadillo

**Suggests** scatterplot3d

**RoxygenNote** 7.1.1

**Encoding** UTF-8

**NeedsCompilation** yes

**Author** Christian Carmona [aut, cre] (<https://orcid.org/0000-0003-0224-4968>), Luis Nieto-Barajas [aut] (<https://orcid.org/0000-0002-0859-7679>), Antonio Canale [ctb] (<https://orcid.org/0000-0002-5403-0040>)

**Maintainer** Christian Carmona <carmona@stats.ox.ac.uk>

**Repository** CRAN

**Date/Publication** 2020-11-30 11:10:06 UTC
meta_param_ex

R topics documented:

- meta_param_ex .......................... 2
- MIXclustering .......................... 3
- plot.MIXcluster ........................ 9
- poverty.data ........................... 10
- summary.MIXcluster .................... 12
- Y_ex_5_1 ............................... 13
- Z_latent_ex_5_1 ......................... 14

Index 17

---

**meta_param_ex**

*Hyper-parameters for testing the BNPMIXcluster package*

**Description**

Values for the parameters used in the three specifications (a), (b) and (c) of the exercise in section 5.1 of the article Carmona et al. (2017).

**Usage**

```r
meta_param_ex
```

**Format**

A data frame with 3 rows and 13 columns.

**Details**

A data frame with 3 rows and 13 columns. Each column is a parameter used in `MIXclustering`.

**See Also**

`MIXclustering`, `Y_ex_5_1`
Description

MIXclustering is used to perform cluster analysis of individuals using a Bayesian nonparametric mixture model that jointly models mixed scale data and accommodates for different sampling probabilities. The model is described in Carmona, C., Nieto-Barajas, L. E., Canale, A. (2016).

Usage

MIXclustering(
  Y, 
  var_type, 
  n_iter_out = 2000, 
  n_burn = 100, 
  n_thin = 2, 
  a_fix = NULL, 
  alpha = 0.5, 
  d_0_a = 1, 
  d_1_a = 1, 
  b_fix = NULL, 
  d_0_b = 1, 
  d_1_b = 1, 
  eta = 2, 
  d_0_z = 2.1, 
  d_1_z = 30, 
  kappa = 5, 
  delta = 4, 
  d_0_mu = 2.1, 
  d_1_mu = 30, 
  sampling_prob = NULL, 
  expansion_f = NULL, 
  log_file = NULL, 
  keep_param_chains = FALSE
)

Arguments

Y  Matrix or data frame containing the data to be clustered.

var_type  Character vector that indicates the type of variable in each column of x. Three possible types:
  • "c" for continuous variables. It is assumed to be Gaussian-shaped.
  • "o" for ordinal variables (binary and ordered categorical).
  • "m" for nominal variables (non-ordered categorical).
n_iter_out  Number of effective iterations in the MCMC procedure for clustering.
n_burn  Number of iterations discarded as part of the burn-in period at the beginning
        MCMC procedure.
n_thin  Number of iterations discarded for thinning the chain (reducing the autocorrela-
        tion). We keep 1 of every n_thin iterations.
a_fix  A numeric value to set the parameter $a$ in the model. If NULL (default), the
        parameter $a$ is assigned a prior distribution. See details.
alpha  Hyperparameter in the prior distribution of $a$. See details.
d_0_a  Hyperparameter in the prior distribution of $a$. See details.
d_1_a  Hyperparameter in the prior distribution of $a$. See details.
b_fix  A numeric value to set the parameter $b$ in the model. If NULL (default), the
        parameter $b$ is assigned a prior distribution. See details.
d_0_b  Hyperparameter in the prior distribution of $b$. See details.
d_1_b  Hyperparameter in the prior distribution of $b$. See details.
eta  Tuning parameter controlling the proposal in the Metropolis-Hastings step for $b$.
d_0_z  Hyperparameter in the prior distribution of the variance for the latent variables.
        See details.
d_1_z  Hyperparameter in the prior distribution of the variance for the latent variables.
        See details.
kappa  Tuning parameter controlling the proposal in the Metropolis-Hastings step for
        the variance of latent variables.
delta  Tuning parameter controlling the proposal in the Metropolis-Hastings step for
        the correlation of latent variables.
d_0_mu  Hyperparameter in the prior distribution of the variance of the location in each
        cluster. See details.
d_1_mu  Hyperparameter in the prior distribution of the variance of the location in each
        cluster. See details.
sampling_prob  vector with the sampling probabilities $\pi_i$ for each individual in case that the data
        come from a complex survey sample. By default $\pi_i = 1$.
expansion_f  vector with the expansion factors, the reciprocal of the sampling probabilities,
        $w_i = 1/\pi_i$. If both sampling_prob and expansion_f are specified, preference
        is given to sampling_prob.
log_file  Specifies a file to save the details with the execution time and the parameters
        used.
keep_param_chains  Indicates if the simulations of parameters $a$, $b$, lambda and omega should be
        returned as output.

Details

The model consists on a Bayesian non-parametric approach for clustering that is capable to com-

bine different types of variables through the usage of associated continuous latent variables. The
clustering mechanism is based on a location mixture model with a Poisson-Dirichlet (PD) process prior on the location parameters \( \mu_i; i = 1, \ldots, n \) of the associated latent variables.

Computational inference about the cluster allocation and the posterior distribution of the parameters are performed using MCMC.


The model consider an individual \( y_i \) that is characterized by a multivariate response of dimension \( p \), i.e., \( y_i = (y_{i,1}, \ldots, y_{i,p}) \). The total number of variables \( p \) is divided into \( c \) continuous variables, \( o \) ordinal variables, and \( m \) nominal variables such that \( p = c + o + m \).

For the continuous variables, it is convenient that the variables have a real support. The user may have transformed the original values before using the function MIXclustering.

For each response \( y_i = (y_{i,1}, \ldots, y_{i,p}) \) (of dimension \( p \)) a corresponding latent vector \( z_i = (z_{i,1}, \ldots, z_{i,q}) \) (of dimension \( q \)) is created, according to the following:

- For each continuous variable \( y_{i,j}; j = 1, \ldots, c \) the algorithm uses a latent with the same values \( z_{i,j} = y_{i,j} \).
- For each ordinal variable \( y_{i,j}, j = c + 1, \ldots, c + o \), with \( K_j \) different ordered values, the algorithm creates one latent \( z_{i,j} \), that allows to map the categories into continuous values divided by thresholds. For example, for a binary \( y_j \), we have \( y_j = 0 \) if \( z_j < 0 \) and \( y_j = 1 \) if \( z_j > 0 \).
- For each nominal variable \( y_{i,j}, j = c + o + 1, \ldots, c + o + m \), with \( L_j \) categories, the algorithm require \( L_j - 1 \) latent variables, whose relative order is consistent with the observed category.

The data may come from a complex survey sample where each individual \( y_i \) has known sampling probability \( \pi_i, i = 1, \ldots, n \). The reciprocal of these sampling probabilities, \( w_i = 1/\pi_i \), are called expansion factors or sampling design weights.

The joint model for the latent vector is therefore:

\[
(z_i|\mu_i, \Sigma) \sim N_q(\mu_i, \pi_i \Sigma)
\]

(Note: the final model in Carmona et al. (2016) has variance \( \kappa \pi_i \Sigma \). This value of \( \kappa \) can be used in the package through a transformed sampling probability vector \( \pi_i^* = \kappa \pi_i \).)

The clustering model will be based in an appropriate choice of the prior distribution on the \( \mu_i \)'s. A clustering of the \( \mu_i \)'s will induce a clustering of the \( y_i \)'s. Our prior on the \( \mu_i \)'s will be:

\( \mu_i \sim G - \text{G}, \text{iid for } i = 1, \ldots, n \)

Where \( G \sim PD(a, b, G_0) \) is a Poisson-Dirichlet process with parameters \( a \in [0, 1), b > -a \) and centering measure \( G_0 \). The Dirichlet and the normalized stable processes arise when \( a = 0 \) and when \( b = 0 \), respectively.

In consequence, this choice of prior implies that the \( \mu_i \)'s are exchangeable with marginal distribution \( \mu_i \sim G - \text{G}_0 \) for all \( i = 1, \ldots, n \).

In our case, \( G(\mu) = N(0, \Sigma_\mu) \), where \( \Sigma_\mu = \text{diag}(\sigma_{\mu_1}^2, \ldots, \sigma_{\mu_q}^2) \).

The parameters \( a \) and \( b \) in the model define the PD process and therefore control the number of groups. These parameters can be fixed, resulting in a larger/smaller number of groups if assigned a larger/smaller value, respectively.

There are 9 hyperparameters in the function that also characterize the prior distributions in the model.
• \( f(a) = \alpha \cdot I(a=0) + (1-\alpha) \cdot \text{dbeta}(a \mid d_{0 \cdot a}, d_{0 \cdot a}) \)
• \( f(b \mid a) = \text{dgamma}(b + a \mid d_{0 \cdot b}, d_{1 \cdot b}) \)
• \( \sigma^2 \sim \text{inverse-gamma}(d_{0 \cdot z}, d_{1 \cdot z}) \)
• \( \sigma^2_{\mu} \sim \text{inverse-gamma}(d_{0 \cdot \mu}, d_{1 \cdot \mu}) \)

The definition of these values also affect the number of resulting clusters since they affect the variance implied in the model.

For example, increasing the values of \( d_{1 \cdot a} \) and \( d_{1 \cdot b} \) reduce the number of groups.

Finally, the function parameters \( \eta, \kappa, \delta \) are tuning parameters that control the acceptance rate in the random-walk MH steps of the new proposed values for the parameters \( b, \Lambda_{i,j}, \Omega_{i,j} \) (variance of latents) and \( \Omega_{i,j} \) (correlation of latents). These parameters are not recommended to be changed (used in the internal functions: sampling\_b, sampling\_Lambda\_jj, sampling\_Omega\_ij).

Value

\textbf{MIXclustering} returns a S3 object of class "MIXcluster".

The generic methods \texttt{summary} and \texttt{plot} are defined for this class.

An object of class "MIXcluster" is a list containing the following components:

- \texttt{cluster} vector with the cluster allocation for each row in the data. It corresponds to the iteration which is Closest-To-Average (CTA) arrangement.
- \texttt{cluster\_heterogeneity} Heterogeneity Measure (HM) for the cluster in the previous point. The HM measure is discussed in section 4 of Carmona et al. (2017).
- \texttt{Y.cluster\_summary} a summary of the data divided by the allocation in \$cluster.
- \texttt{Y.var\_type} vector with the variable types in the data.
- \texttt{Y.na} vector specifying the rows with missing values.
- \texttt{Y.n} number of rows in the data.
- \texttt{Y.p} number of variables in the data.
- \texttt{MC.clusters} matrix with the cluster allocation for each row in the data. Each column corresponds to an effective iteration in the MCMC simulation of the model (after discarding burn-in and thinning iterations).
- \texttt{MC.clusters\_heterogeneity} Heterogeneity Measure (HM) for all the clusters returned in \texttt{MC.clusters}.
- \texttt{cluster.matrix\_avg} average similarity matrix of size \( n \) by \( n \).
- \texttt{MC.values} a list with the simulated values of the chains for the parameters \( a,b,\Lambda,\Omega \).
- \texttt{MC.accept\_rate} a named vector with the acceptance rates for each parameter. It includes iterations that are discarded in the burn-in period and thinning.
- \texttt{call} the matched call.

References

See Also

*summary.MIXcluster* for a summary of the clustering results, *plot.MIXcluster* for graphical representation of results.

Examples

```r
# Simulation study 1  #
# Carmona et al. (2017)  #

# Data and parameters are discussed in section 5.1 of Carmona et al. (2017) #

# Set seed for reproducibility #
set.seed(0)

# Specification of data Y #
help(Y_ex_5_1)

# Observable data #
# Choose scenario: 1, 2, or 3
ex_i <- 1

# Prior specification #
# Choose "a", "b" or "c"
param_j <- "c"

# Specify the data type that is being provided to the method
var_type_Y_ex_5_1 <- list(c("c","c","c"),
c("o","o"),
c("o","o","o","c")
)

## Not run:  
cluster_ex <- MIXclustering( Y = as.matrix(Y_ex_5_1[[ ex_i ]]),
 var_type=var_type_Y_ex_5_1[[ ex_i ]],
   n_iter_out=1500,
   n_burn=200,
   n_thin=3,
   alpha = meta_param_ex[ param_j, "alpha" ],
   d_0_a = meta_param_ex[ param_j, "d_0_a" ],
   d_1_a = meta_param_ex[ param_j, "d_1_a" ],
   d_0_b = meta_param_ex[ param_j, "d_0_b" ],
   d_1_b = meta_param_ex[ param_j, "d_1_b" ],
   eta = meta_param_ex[ param_j, "eta" ],
   kappa = meta_param_ex[ param_j, "kappa" ],
   delta = meta_param_ex[ param_j, "delta" ],
   d_0_z = meta_param_ex[ param_j, "d_0_z" ],
   d_1_z = meta_param_ex[ param_j, "d_1_z" ],
```

d_0_mu = meta.param.ex[ param_j, "d_0_mu" ],
d_1_mu = meta.param.ex[ param_j, "d_1_mu" ]

# Summary of clustering results
summary(cluster_ex)

# Visualizing clustering results
plot(cluster_ex,type="heatmap")
plot(cluster_ex,type="chain")

# Comparison of cluster configurations #
# 1) Minimum distance with average MCMC iterations
# 2) Minimum Heterogeneity Measure (HM)
plot( x=jitter(cluster_ex$cluster),y=jitter(cluster_ex$clusterHMmin), col="#FF000080", pch=20,
     main=paste("Comparison of two relevant cluster configurations"),
     xlab="minimizes distance to average MCMC grouping", ylab="minimizes Heterogeneity Measure" )

# Comparison with the original clusters in the simulated data
plot(x=jitter(Z_latent_ex_5_1$cluster),
     y=jitter(cluster_ex$cluster),
     main=paste("Comparison real configuration with the model results"),
     xlab="Real cluster",
     ylab="Model cluster",
     pch=19, col="#FF000080")

## End(Not run)

########################################################################
# Households data #
# Carmona et al. (2017) #
########################################################################

# Testing "MIXclustering" function with poverty.data #
# Data and parameters are discussed in section 5.3 of Carmona et al. (2017) #

# Set seed for reproducibility #
set.seed(0)

## Not run:
# relevant variables for clustering households #
Y.names <- c("ict_norm",
             "ic_ali","ic_asalud","ic_cv",
             "ic_rezedu","ic_sbv","ic_segsoc",
             "niv_ed","tam_loc")
Y.var.type <- c("c","o","o","o","o","o","o","o")

# using only data from state 15 (Edomex) #
aux_subset <- rep(TRUE,nrow(poverty.data))
aux_subset <- aux_subset & is.element(substr(poverty.data$folioviv,1,2),"15")

Y.data <- poverty.data[aux_subset,Y.names]

### Sampling probability dependin on the scenario ###
plot.MIXcluster

# Scenario description in section 5.3 of Carmona et al. (2017) #
# Choose 1, 2 or 3 #
poverty_sampling_spec <- 3

if (poverty_sampling_spec == 1) {
  k <- 1
  sampling_prob_pov <- rep(1,nrow(Y_data))
} else if (poverty_sampling_spec == 2) {
  k <- 2 * mean(poverty.data[aux_subset,"factor_hog"])
  sampling_prob_pov <- 1/poverty.data[aux_subset,"factor_hog"]
} else if (poverty_sampling_spec == 3) {
  k <- 4 * mean(poverty.data[aux_subset,"factor_hog"])
  sampling_prob_pov <- 1/poverty.data[aux_subset,"factor_hog"]
}

cluster_poverty <- MIXclustering( Y=Y_data,
  var_type=Y_var_type,
  n_iter_out=1500,
  n_burn=200,
  n_thin=3,

  alpha = 0.5,
  d_0_a = 1, d_1_a = 1,
  d_0_b = 1, d_1_b = 1,

  eta = 2,
  kappa = 5,
  delta = 4,

  d_0_z = 2.1, d_1_z = 30,
  d_0_mu = 2.1, d_1_mu = 30,

  sampling_prob = k * sampling_prob_pov )

summary(cluster_poverty)
plot(cluster_poverty,type="heatmap")
plot(cluster_poverty,type="chain")

## End(Not run)

---

plot.MIXcluster  

Plotting clustering results for "MIXcluster" objects

Description

Plotting method for objects inheriting from class "MIXcluster".
Usage

```r
## S3 method for class 'MIXcluster'
plot(
  x,
  type = c("heatmap", "chain")[1],
  chain.obj = c("n.cluster", "a", "b", "Lambda", "Omega", "all")[1],
  ...)
```

Arguments

- **x**: an object of class "MIXcluster"
- **type**: what type of plot should be drawn. Possible types are:
  - "heatmap" (default) draws a heatmap of the average similarity matrix for the effective iterations of the MCMC.
  - "chain" for the evolution and histograms of the chains for parameters in the model.
- **chain.obj**: if type="chain", this specifies what chain will be plotted. Possible types are:
  - "n.cluster" (default) for the number of clusters.
  - "a" for the $a$ parameter of the model.
  - "b" for the $b$ parameter of the model.
  - "Lambda" one plot for each element in the diagonal of the $\Lambda$ matrix of the model (variance of latent variables).
  - "Omega" one plot for each element above the diagonal of the $\Omega$ matrix of the model (correlation between latent variables).
  - "all" for all of the above.
- **...**: further arguments passed to or from other methods.

See Also

MIXclustering

---

**poverty.data**

*Poverty data for testing the BNPMIXcluster package*

Description

Poverty indicators observed in Mexico for 2014.

The original data is available in the file "R_2014.zip" from CONEVAL's website: [http://www.coneval.org.mx/Medicion/MP/Paginas/Programas_BD_10_12_14.aspx](http://www.coneval.org.mx/Medicion/MP/Paginas/Programas_BD_10_12_14.aspx)


This data frame presents indicators aggregated by household. The aggregation was done by the authors according with code in section Examples.
poverty.data

Usage

poverty.data

Details

poverty.data is a data frame with 58121 rows and 13 variables, with the following columns:

proyecto  Data source identifier (1=MCS, 2=ENIGH)
folioviv  Household identifier level 1
foliohog  Household identifier level 2
ict_norm  (continuous) Total income in the household (in Mexican pesos).
ic_alii  (binary) Indicator for deprivation to feeding: 1-yes,0-no
ic_asalud  (binary) Indicator for deprivation to health services: 1-yes,0-no
ic_cv  (binary) Housing quality: 1-bad, 0-good
ic_rezedu  (binary) Indicator for education backwardness: 1-yes,0-no
ic_sbv  (binary) Indicator for deprivation to basic public services: 1-yes,0-no
ic_segsoc  (binary) Indicator for deprivation to social security: 1-yes,0-no
niv_ed  (categorical, ordered) Maximum educational level in the household: 0-incomplete primary: 1-incomplete secondary, 2-complete secondary or more
tam_loc  (categorical, nominal) Size of locality according to the number of people living there: 1-100000, 2-[15000, 100000), 3-[2500, 15000), 4-[0, 2500)
factor_hog  Expansion factor for the household, according to the complex survey design.

See Also

MIXclustering

Examples

##### Generates poverty.data using the original data from CONEVAL's website #####

## Not run:

# step 1:
# Download and unzip the file "R_2014.zip"
# available in:
# http://www.coneval.org.mx/Medicion/MP/Documents/Programas_calculo_pobreza_10_12_14/R_2014.zip

coneval.poverty.data <- read.csv("pobreza_14.csv", na.strings=c("NA",""))

# step 3:
# Execute the following code...

var_id <- c("proyecto","folioviv","foliohog","numren")
for(i in match(var_id,colnames(coneval.poverty.data))) {
    coneval.poverty.data[,i] <- formatC( x=as.numeric(coneval.poverty.data[,i]),
                                         width=max(nchar(coneval.poverty.data[,i])),
                                         format="f",flag="0",digits=0
    }
}

# normalizing the continuous variable for income #
b <- quantile(coneval.poverty.data$ict,probs=0.01)
coneval.poverty.data$ict_norm <- log(coneval.poverty.data$ict+b)

# Aggregating data at household level
Y_names <- c("ict_norm","ic_all","ic_asalud","ic_cv",
            "ic_rezedu","ic_sbv","ic_segsoc",
            "niv_ed","tam_loc")
agg_form <- as.formula("cbind("paste(c(Y_names,"factor_hog"),collapse="","","
                                "proyecto+folioviv+foliohog"
                        
poverty.data <- aggregate(agg_form,FUN="max",data=coneval.poverty.data)

## End(Not run)

summary.MIXcluster

Summary of clustering results

Description

summary method for class "MIXcluster".

Usage

### S3 method for class 'MIXcluster'
summary(object, ...)

Arguments

- **object**: an object of class "MIXcluster"
- **...**: further arguments passed to or from other methods.

See Also

- MIXclustering
Y_ex_5_1

Simulated data for testing the BNPMIXcluster package

Description

List with three data frames. Each dataset consists of the data \( Y_i \) described in the exercise of section 5.1 in the article Carmona et al (2017).

The data \( Y_{\text{ex}_5_1} \) is a transformation of the simulated data \( Z_{\text{latent}_5_1} \).

Usage

\( Y_{\text{ex}_5_1} \)

Format

A list with three data frames.

Details

A list with three data frames. Each data frame with 100 rows.

See Also

MIXclustering

Examples

### Show the relation between \( Y_{\text{ex}_5_1} \) and \( Z_{\text{latent}_5_1} \) ###

```r
plot(y=Y_ex_5_1[[3]][,"Y1"],x=Z_latent_ex_5_1$Z1,pch=20,col=2); abline(v=c(5),lty=3)
plot(y=Y_ex_5_1[[3]][,"Y2"],x=Z_latent_ex_5_1$Z2,pch=20,col=2); abline(v=c(5),lty=3)
plot(y=Y_ex_5_1[[3]][,"Y3"],x=Z_latent_ex_5_1$Z3,pch=20,col=2); abline(v=c(5),lty=3)
```

##############################
# Exercise 5.1
# Data definition
##############################

### Code to generate \( Y_{\text{ex}_5_1} \) from \( Z_{\text{latent}_5_1} \) ###

```r
Y_ex_5_1 <- list()

# (I) #
# Three continuous variables (Y1, Y2, Y3)
# defined as Yi = Z_i, for i=1, 2, 3.
Y_ex_5_1[[1]] <- Z_latent_ex_5_1[,c("Z1","Z2","Z3")]
```

Y_ex_5_1
## (II) ##

### (II) ###

```
# two binary variables (Y_1, Y_3) defined as
# Y_1 = I(Z_1 > 5)
# Y_3 = I(Z_3 > 3)
Y_ex_5_1_i <- data.frame(matrix(NA,nrow=nrow(Z_latent_ex_5_1),ncol=2))
colnames(Y_ex_5_1_i) <- paste("Y",c(1,3),sep="")
Y_ex_5_1_i$Y1 <- findInterval( Z_latent_ex_5_1$Z1, c(-Inf,5,Inf) )-1
Y_ex_5_1_i$Y3 <- findInterval( Z_latent_ex_5_1$Z3, c(-Inf,3,Inf) )-1
Y_ex_5_1[[2]] <- Y_ex_5_1_i
```

### (III) ##

```
# two binary variables (Y_1, Y_3) defined as in Scenario (II)
# one ordinal variable Y_2 such that Y_2 = I(4 < Z_2 < 5) + 2 * I(Z_2 > 5)
# and one continuous variable Y_4 distributed N(0, 1)
Y_ex_5_1_i <- data.frame(matrix(NA,nrow=nrow(Z_latent_ex_5_1),ncol=4))
colnames(Y_ex_5_1_i) <- paste("Y",1:4,sep="")
Y_ex_5_1_i$Y1 <- Y_ex_5_1[[2]]$Y1
Y_ex_5_1_i$Y2 <- findInterval( Z_latent_ex_5_1$Z2, c(-Inf,4,5,Inf) )-1
Y_ex_5_1_i$Y3 <- Y_ex_5_1[[2]]$Y3
Y_ex_5_1_i$Y4 <- rnorm(n=nrow(Z_latent_ex_5_1),mean=0,sd=1)
Y_ex_5_1[[3]] <- Y_ex_5_1_i
```

Y_ex_5_1
```

---

Z_latent_ex_5_1  

*Simulated data for testing the BNPMIXcluster package*

---

**Description**

Simulated values for three continuous variables under the existence of three clusters.

The data consists of a three-variate Normal distribution with different mean and covariance matrix between clusters.

This can be assumed either as continuous data to be clustered Y=(Y_1,Y_2,Y_3); or also can be used as the underlying latent data that can be transformed into observable variables Y_i=f(Z_i), which can be continuous or categorical.

**Usage**

Z_latent_ex_5_1

**Format**

A data frame with 100 rows and 4 variables.

- **cluster** Indicates the cluster for each row
- **Z1,Z2,Z3** Continuous values coming from a multivariate normal distribution, given the cluster
Details

A data frame with 100 rows and 4 variables.

See Also

MIXclustering

Examples

### Visualizing the simulated data for clustering ###

```r
require(scatterplot3d)

cluster_color <- c(rgb(1,0,0,alpha = 0.5),
   rgb(0,0,1,alpha = 0.5),
   rgb(0,0.5,0,alpha = 0.5))
cluster_color <- cluster_color[Z_latent_ex_5_1$cluster]
cluster_pch <- c(19,15,17)[Z_latent_ex_5_1$cluster]
par(mfrow=c(2,2))
par(mar=c(4,5,2,2))
scatterplot3d::scatterplot3d(x=Z_latent_ex_5_1$Z1,y = Z_latent_ex_5_1$Z2, z=Z_latent_ex_5_1$Z3,
   color=cluster_color,pch=cluster_pch,
   xlab="Z1",ylab="Z2",zlab="Z3",
   main="Simulated data in 3 clusters"
)
par(mar=c(4,5,2,2))
plot(Z_latent_ex_5_1[,c("Z2","Z3")],col=cluster_color,pch=cluster_pch,xlab="Z2",ylab="Z3")
par(mar=c(4,5,2,2))
plot(Z_latent_ex_5_1[,c("Z1","Z3")],col=cluster_color,pch=cluster_pch,xlab="Z1",ylab="Z3")
par(mar=c(4,5,2,2))
plot(Z_latent_ex_5_1[,c("Z1","Z2")],col=cluster_color,pch=cluster_pch,xlab="Z1",ylab="Z2")
```

# Exercise 5.1

### Code to generate the simulated data from scratch ###

```r
require(MASS)
set.seed(0)
n.sim <- 100
n.cluster <- 3
p <- 3
mu_Z_latent <- matrix( c( 2 , 2 , 5 ,
   6 , 4 , 2 ,
   1 , 6 , 2 ),
   nrow=3,ncol=3)
```
nrow=n.cluster, ncol=p, byrow=TRUE)

sigma_Z_latent <- array(dim=c(3,3,3))
sigma_Z_latent[,,1] <- diag(3)
sigma_Z_latent[,,2] <- matrix( c( 0.1 , 0 , 0 , 0 , 2 , 0 , 0 , 0 , 0.1 ),
  nrow=n.cluster, ncol=p, byrow=TRUE)
sigma_Z_latent[,,3] <- matrix( c( 2 , 0 , 0 , 0 , 0.1 , 0 , 0 , 0 , 0.1 ),
  nrow=n.cluster, ncol=p, byrow=TRUE)

Z_cluster <- data.frame(cluster=sample(x=1:n.cluster,size=n.sim,replace=TRUE))

Z_latent <- matrix(NA,nrow=n.sim,ncol=p)
for( i in unique(Z_cluster$cluster) ) {
  Z_latent[Z_cluster[,1]==i,] <- MASS::mvrnorm( n=sum(Z_cluster[,1]==i),
    mu=mu_Z_latent[i,],
    Sigma=sigma_Z_latent[,,i] )
}
colnames(Z_latent) <- paste("Z",1:ncol(Z_latent),sep="")
Z_latent_ex_5_1 <- cbind(Z_cluster,Z_latent)
Z_latent_ex_5_1
Index

meta_param_ex. 2
MIXclustering, 2, 3, 10–13, 15

plot. 6
plot.MIXcluster, 7, 9
poverty.data, 10

summary, 6
summary.MIXcluster, 7, 12

Y_ex_5_1, 2, 13

Z_latent_ex_5_1, 14