Package ‘dmbc’

April 26, 2022

Type  Package
Title  Model Based Clustering of Binary Dissimilarity Measurements
Version  1.0.1
Date  2022-04-26
Description  Functions for fitting a Bayesian model for grouping binary dissimilarity matrices in homogeneous clusters. Currently, it includes methods only for binary data (<doi:10.18637/jss.v100.i16>).
Author  Sergio Venturini [aut, cre], Raffaella Piccarreta [ctb]
Maintainer  Sergio Venturini <sergio.venturini@unicatt.it>
License  GPL (>= 2)
NeedsCompilation  yes
Repository  CRAN
LazyData  TRUE
Imports  abind, bayesplot (>= 1.7.0), coda (>= 0.19-3), ggplot2 (>= 3.2.1), ggrepel (>= 0.8.1), graphics, MCMCpack (>= 1.4-4), modeltools (>= 0.2-22), parallel (>= 3.6.1), robustbase (>= 0.93-5), robustX (>= 1.2-5), stats4 (>= 3.6.0), tools
Suggests  knitr, mcmcplots, rlecuyer, testthat
Depends  methods, R (>= 3.6.0), stats, utils
LinkingTo  Rcpp, RcppArmadillo, RcppProgress
BugReports  https://github.com/sergioventurini/dmbc/issues
Encoding  UTF-8
RoxygenNote  7.1.2
Date/Publication  2022-04-26 10:10:02 UTC
# R topics documented:

- `dmbc-package` ................................................. 3
- `adjust_x` ..................................................... 4
- `animals` ...................................................... 5
- `bmds` .......................................................... 6
- `bmds_get_x_mode` ........................................... 10
- `check_list_na` ................................................. 11
- `clusters,dmbc_config-method` ................................ 12
- `comp_ssr` ...................................................... 13
- `dmbc` .......................................................... 14
- `dmbc_check_groups` ........................................... 16
- `dmbc_config-class` ............................................ 17
- `dmbc_control` .................................................. 18
- `dmbc_data-class` .............................................. 20
- `dmbc_fit` ....................................................... 21
- `dmbc_fit-class` ............................................... 23
- `dmbc_fit_list-class` ......................................... 24
- `dmbc_fit_list_to_list` ....................................... 25
- `dmbc_fit_list_to_mcmc.list` ................................ 26
- `dmbc_fit_to_mcmc` ............................................ 27
- `dmbc_get_configuration` ..................................... 28
- `dmbc_get_map` .................................................. 29
- `dmbc_get_ml` ................................................... 31
- `dmbc_get_postmean` .......................................... 32
- `dmbc_get_postmedian` ........................................ 34
- `dmbc_IC` ........................................................ 35
- `dmbc_ic-class` ................................................ 37
- `dmbc_init` ...................................................... 38
- `dmbc_logLik` ................................................... 39
- `dmbc_logLik_rbmds` .......................................... 40
- `dmbc_match_groups` ........................................... 41
- `dmbc_mcmc` ..................................................... 42
- `dmbc_model-class` ............................................. 44
- `dmbc_prior` ..................................................... 45
- `initialize,dmbc_config-method` ............................ 46
- `initialize,dmbc_data-method` ................................ 48
- `initialize,dmbc_fit-method` ................................ 48
- `initialize,dmbc_fit_list-method` .......................... 50
- `initialize,dmbc_ic-method` .................................. 51
- `initialize,dmbc_model-method` .............................. 52
- `kinship` ........................................................ 52
- `mdsic` .......................................................... 54
- `plot,dmbc_config,ANY-method` .............................. 55
- `plot,dmbc_data,ANY-method` ................................ 56
- `plot,dmbc_fit,ANY-method` ................................... 57
- `plot,dmbc_fit_list,ANY-method` ............................. 58
- `plot,dmbc_ic,ANY-method` ................................... 59
**Description**

The *dmbc* package implements a Bayesian algorithm for clustering a set of dissimilarity matrices within a model-based framework. In particular, we consider the case where $S$ matrices are available, each describing the dissimilarities among $n$ objects, possibly expressed by $S$ subjects (judges), or measured under different experimental conditions, or with reference to different characteristics of the objects themselves. Specifically, we focus on binary dissimilarities, taking values 0 or 1 depending on whether or not two objects are deemed as similar, with the goal of analyzing such data using multidimensional scaling (MDS). Differently from the standard MDS algorithms, we are interested in partitioning the dissimilarity matrices into clusters and, simultaneously, to extract a specific MDS configuration for each cluster. The parameter estimates are derived using a hybrid Metropolis-Gibbs Markov Chain Monte Carlo algorithm. We also include a BIC-like criterion for jointly selecting the optimal number of clusters and latent space dimensions.

For efficiency reasons, the core computations in the package are implemented using the C programming language and the *RcppArmadillo* package.

The *dmbc* package also supports the simulation of multiple chains through the support of the *parallel* package.

Plotting functionalities are imported from the nice *bayesplot* package. Currently, the package includes methods for binary data only. In future releases routines will be added specifically for continuous (i.e. normal), multinomial and count data.

**dmbc classes**

The *dmbc* package defines the following new classes:

- *dmbc_data*: defines the data to use in a DMBC model.
• **dmbc_model**: defines a DMBC model.
• **dmbc_fit**: defines the results of a DMBC analysis for a single MCMC chain.
• **dmbc_fit_list**: defines the results of a DMBC analysis for multiple MCMC chains.
• **dmbc_ic**: defines the results of the computation of the information criterion for a DMBC analysis.
• **dmbc_config**: defines the estimate of the latent configuration for a DMBC analysis.

The package includes print, summary and plot methods for each one of these classes.

**Resources**

• **Bug reports**: If you have noticed a bug that needs to be fixed, please let us know at the [dmbc issue tracker](https://github.com/sergioventurini/dmbc/issues/).
• **General questions and help**: To ask a question about [dmbc](https) send and email to: <sergio.venturini@unicatt.it>.

**References**


**See Also**

-themedefault for the default ggplot theme used by bayesplot.
-bayesplot-colors to set or view the color scheme used for plotting with bayesplot.
-ggsave in ggplot2 for saving plots.

---

**adjust_x**: Adjustment of the center and orientation of a latent configuration.

**Description**

adjust_x adjusts the center and orientation of a latent configuration in Bayesian (metric) multidimensional scaling (BMDS).

**Usage**

adjust_x(x)

**Arguments**

x Numeric matrix containing the latent configuration.
Value

A list with elements:

- x  A real matrix containing the adjusted latent configuration.
- Sig_x  The variance and covariance matrix of the adjusted latent configuration.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

See Also

bmds for (one-way) Bayesian (metric) multidimensional scaling.

Examples

```r
n <- 100
nr <- 20
nc <- floor(n/nr)
x <- matrix(rnorm(1:n), nrow = nr, ncol = nc)
adj_x <- adjust_x(x)
adj_x$x
adj_x$Sig_x
```

animals  List of binary dissimilarity matrices among 18 animals.

Description

To illustrate the MDS analysis of sorting data, Takane et al. (2009) refer to judgments on the similarity between $n = 18$ animals expressed by $S = 20$ subjects. Each subject was asked to divide the animals into as many groups as needed, based on their similarity. We converted these values to 0 or 1 depending on whether a pair of animals is placed or not in the same group by a subject.

Usage

data(animals)

Format

A `dmbc_data` object whose `diss` element is a list of 20 binary dissimilarity matrices. Each matrix is defined as a `dist` object measuring whether each pair of the 18 animals has is placed in the same group (1) or not (0).

The `dist` objects have rows and columns that are named as follows:

- **be bear**
- **cm camel**
ct cat
cw cow
dg dog
el elephant
gf giraffe
fx fox
hs horse
li lion
mk monkey
ms mouse
pg pig
rb rabbit
sh sheep
sq squirrel
tg tiger
wf wolf

References

Examples

data(animals)
library(bayesplot)
cols <- color_scheme_set("teal")
plot(animals, colors = unlist(cols)[c(1, 6)], font = 1, cex.font = 0.75)

bmds

Bayesian multidimensional scaling (BMDS) using Markov Chain Monte Carlo (MCMC).

Description

bmds computes the Bayesian multidimensional scaling (BMDS) solutions using Markov Chain Monte Carlo for a range of specified latent space dimensions.
Usage

bmds(
  D,
  min_p = 1,
  max_pm1 = 6,
  burnin = 0,
  nsim = 13000,
  ic = TRUE,
  verbose = TRUE
)

Arguments

D Observed dissimilarities (provided as a distance matrix).
min_p A length-one numeric vector providing the minimum value of the latent space dimension to use.
max_pm1 A length-one numeric vector providing the maximum value of the latent space dimension to use (minus 1).
burnin A length-one numeric vector providing the number of iterations to use for burnin.
nsim A length-one numeric vector providing the number of iterations to use in the MCMC simulation after burnin.
ic Logical scalar. If TRUE computes the MDS information criterion (MDSIC) for all solution requested.
verbose Logical scalar. If TRUE prints information regarding the evolution of the simulation.

Value

A list with the following elements:

x.chain MCMC chain of the latent configuration coordinates.
sigma.chain MCMC chain of the random error.
lambda.chain MCMC chain of the latent configuration variances.
stress Numeric vector of the stress function values.
mdsIC List with two elements, the MDSIC and BIC values for the required solutions.
accept Numeric matrix of acceptance rates.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

References

See Also
cmdscale for classical (metric) multidimensional scaling.

Examples

```
## Not run:  
# Airline Distances Between Cities  
airline <- read.csv(file = system.file("extdata", "airline.csv",  
    package = "dmbc"))  
airline.nm <- airline[, 1]  
airline <- airline[, 2:31]  
colnames(airline) <- airline.nm  
airline <- as.dist(airline)  
min_p <- 1  
max_p <- 4  
burnin <- 200  
nsim <- 1000  
totiter <- burnin + nsim  
  
airline.mds <- cmdscale(airline, max_p)  
airline.bmds <- bmds(airline, min_p, max_p, burnin, nsim)  
  
opar <- par(mfrow = c(1, 2))  
plot(min_p:max_p, airline.bmds$mdsIC$mdsic, type = "b",  
    main = "MDS Information Criterion", xlab = "p", ylab = "MDSIC")  
MDSICmin <- which.min(airline.bmds$mdsIC$mdsic)  
points((min_p:max_p)[MDSICmin], airline.bmds$mdsIC$mdsic[MDSICmin],  
    col = "red", pch = 10, cex = 1.75, lwd = 1.5)  
airline.bmds.x.mode <- bmds_get_x_mode(airline, airline.bmds, MDSICmin,  
    min_p, max_p, start = (burnin + 1), end = totiter)  
airline.bmds.d <- dist(airline.bmds.x.mode)  
airline.mds.d <- dist(airline.mds[, 1:(min_p:max_p)[MDSICmin]])  
plot(airline, airline.bmds.d, type = "n", xlab = "observed",  
    ylab = "estimated", main = "Airline Distances \n Between Cities",  
    xlim = c(0, max(airline, airline.bmds.d)),  
    ylim = c(0, max(airline, airline.bmds.d)))  
abline(0, 1, lty = 2, col = "gray")  
points(airline, airline.mds.d, pch = 19, col = "cyan", cex = .5)  
points(airline, airline.bmds.d, pch = 19, col = "magenta", cex = .5)  
legend(x = "bottomright", legend = c("Classical MDS", "Bayesian MDS"),  
    pch = c(19, 19), col = c("cyan", "magenta"))  
par(opar)  
  
# Careers of Lloyds Bank Employees  
lloyds <- read.csv(file = system.file("extdata", "lloyds.csv",  
    package = "dmbc"))  
lloyds.nm <- lloyds[, 1]  
lloyds <- lloyds[, 2:81]  
colnames(lloyds) <- lloyds.nm  
lloyds <- as.dist(lloyds)```
min_p <- 1
max_p <- 12
burnin <- 200
nsim <- 1000
totiter <- burnin + nsim

lloyds.mds <- cmdscale(lloyds, max_p)
lloyds.bmds <- bmds(lloyds, min_p, max_p, burnin, nsim)

opar <- par(mfrow = c(1, 2))
plot((min_p:max_p), lloyds.bmds$mdsIC$mdsic, type = "b",
    main = "MDS Information Criterion", xlab = "p", ylab = "MDSIC")
MDSICmin <- which.min(lloyds.bmds$mdsIC$mdsic)
points((min_p:max_p)[MDSICmin], lloyds.bmds$mdsIC$mdsic[MDSICmin],
    col = "red", pch = 10, cex = 1.75, lwd = 1.5)
lloyds.bmds.x.mode <- bmds_get_x_mode(lloyds, lloyds.bmds, MDSICmin,
    min_p, max_p, start = (burnin + 1), end = totiter)
lloyds.bmds.d <- dist(lloyds.bmds.x.mode)
lloyds.mds.d <- dist(lloyds.mds[, 1:(min_p:max_p)[MDSICmin]])
plot(lloyds, lloyds.mds.d, type = "n", xlab = "observed",
    ylab = "estimated", main = "Careers of Lloyds \n Bank Employees, 1905-1950",
    xlim = c(0, max(lloyds, lloyds.mds.d)),
    ylim = c(0, max(lloyds, lloyds.bmds.d)))
abline(0, 1, lty = 2, col = "gray")
points(lloyds, lloyds.mds.d, pch = 19, col = "cyan", cex = .5)
points(lloyds, lloyds.bmds.d, pch = 19, col = "magenta", cex = .5)
legend(x = "topleft", legend = c("Classical MDS", "Bayesian MDS"),
    pch = c(19, 19), col = c("cyan", "magenta"))
par(opar)

# Road distances (in km) between 21 cities in Europe
data(eurodist, package = "datasets")
min_p <- 1
max_p <- 10
burnin <- 200
nsim <- 1000
totiter <- burnin + nsim
eurodist.mds <- cmdscale(eurodist, max_p)
eurodist.bmds <- bmds(eurodist, min_p, max_p, burnin, nsim)

opar <- par(mfrow = c(1, 2))
plot((min_p:max_p), eurodist.bmds$mdsIC$mdsic, type = "b",
    main = "MDS Information Criterion", xlab = "p", ylab = "MDSIC")
MDSICmin <- which.min(eurodist.bmds$mdsIC$mdsic)
points((min_p:max_p)[MDSICmin], eurodist.bmds$mdsIC$mdsic[MDSICmin],
    col = "red", pch = 10, cex = 1.75, lwd = 1.5)
eurodist.bmds.x.mode <- bmds_get_x_mode(eurodist, eurodist.bmds,
    MDSICmin, min_p, max_p, start = (burnin + 1), end = totiter)
bmds_get_x_mode

bmds_get_x_mode returns the latent configuration that produced the largest posterior value during the MCMC.

Usage

bmds_get_x_mode(D, res, p.i, min_p, max_p, start, end)

Arguments

D  Observed dissimilarities (provided as a distance matrix).
res Results of a BMDS analysis as obtained with the bmds function.
p.i A length-one numeric vector providing the index of the solution to use.
min_p A length-one numeric vector providing the minimum value of the latent space dimension to use.
max_p A length-one numeric vector providing the maximum value of the latent space dimension to use.
start A length-one numeric vector providing the iteration number to start from.
end A length-one numeric vector providing the iteration number where to end.

Value

A real matrix containing the posterior mode latent configuration.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>
check_list_na

See Also

bmds for (one-way) Bayesian (metric) multidimensional scaling.

Examples

```r
## Not run:
# Airline Distances Between Cities
airline <- read.csv(file = system.file("extdata", "airline.csv", package = "dmbc"))
airline.nm <- airline[, 1]
airline <- airline[, 2:31]
colnames(airline) <- airline.nm
airline <- as.dist(airline)
min_p <- 1
max_p <- 4
burnin <- 200
nsim <- 1000
totiter <- burnin + nsim

airline.mds <- cmdscale(airline, max_p)
airline.bmds <- bmds(airline, min_p, max_p, burnin, nsim)

opar <- par(mfrow = c(1, 2))
plot(min_p:max_p, airline.bmds$mdsIC$mdsic, type = "b",
main = "MDS Information Criterion", xlab = "p", ylab = "MDSIC")
MDSICmin <- which.min(airline.bmds$mdsIC$mdsic)
points((min_p:max_p)[MDSICmin], airline.bmds$mdsIC$mdsic[MDSICmin],
col = "red", pch = 10, cex = 1.75, lwd = 1.5)

airline.bmds.x.mode <- bmds_get_x_mode(airline, airline.bmds, MDSICmin,
min_p, max_p, start = (burnin + 1), end = totiter)
airline.bmds.d <- dist(airline.bmds.x.mode)
airline.mds.d <- dist(airline.mds[, 1:(min_p:max_p)[MDSICmin]])
plot(airline, airline.bmds.d, type = "n", xlab = "estimated",
ylab = "Airline Distances \n Between Cities",
xlim = c(0, max(airline, airline.bmds.d)),
ylim = c(0, max(airline, airline.bmds.d)))
abline(0, 1, lty = 2, col = "gray")
points(airline, airline.mds.d, pch = 19, col = "cyan", cex = .5)
points(airline, airline.bmds.d, pch = 19, col = "magenta", cex = .5)
legend("bottomright", legend = c("Classical MDS", "Bayesian MDS"), pch = c(19, 19), col = c("cyan", "magenta"))
par(opar)

## End(Not run)

check_list_na

Auxiliary function to recursively check NAs in a list.
Description
check_list_na() compares two lists and fills in the missing elements in the first with those included in the second. The comparison is recursive in the sense that the process is repeated for all lists included in those given.

Usage
check_list_na(orig, des)

Arguments
orig A list whose content must be checked.
des A list to use as a reference with which compare the first one.

Value
A list with all elements added.

Author(s)
Sergio Venturini <sergio.venturini@unicatt.it>

Examples
G <- 5
prior <- list(eta = list(a = rep(1, G), b = rep(2, G)))
check_list_na(prior, dmbc_prior())

Description
Extract the final cluster memberships from a dmbc_config class instance.

Usage
## S4 method for signature 'dmbc_config'
clusters(object, newdata = NULL, ...)

Arguments
object An object of class dmbc_config.
newdata An object of no explicit specification (currently ignored).
... Further arguments to pass on (currently ignored).
comp_ssr

Author(s)
Sergio Venturini <sergio.venturini@unicatt.it>

Description
comp_ssr computes the sum of squared residuals (SSR) from the observed distances (diss) and the given latent coordinates (x).

Usage
comp_ssr(x, diss)

Arguments
x
Real matrix containing the latent configuration.
diss
Observed dissimilarities (provided as a distance matrix).

Value
A length-one numeric vector providing the SSR for its arguments.

Author(s)
Sergio Venturini <sergio.venturini@unicatt.it>

See Also
bmds for (one-way) Bayesian (metric) multidimensional scaling.

Examples
n <- 10000
nr <- 200
nc <- floor(n/nr)
x <- matrix(rnorm(1:n), nrow = nr, ncol = nc)
obsdiss <- dist(x)
ssr <- numeric(ncol(x))
for (i in 1:ncol(x)) {
  ssr[i] <- comp_ssr(x[, 1:i], obsdiss)
}
plot(ssr, xlab = "number of dimensions", ylab = "SSR", type = "b")
Estimation of a DMBC model.

**Description**

dmbc(), the main function of the package, estimates a DMBC model for a given set of $S$ dissimilarity matrices.

**Usage**

```r
dmbc(
  data,
  p = 2,
  G = 3,
  control = dmbc_control(),
  prior = NULL,
  cl = NULL,
  post_all = FALSE
)
```

**Arguments**

- **data** An object of class dmbc_data containing the data to analyze.
- **p** A length-one numeric vector indicating the number of dimensions of the latent space.
- **G** A length-one numeric vector indicating the number of cluster to partition the $S$ subjects.
- **control** A list of control parameters that affect the sampling but do not affect the posterior distribution. See dmbc_control() for more details.
- **prior** A list containing the prior hyperparameters. See dmbc_prior() for more details.
- **cl** An optional parallel or snow cluster for use if parallel = "snow". If not supplied, a cluster on the local machine is created for the duration of the dmbc() call.
- **post_all** A length-one logical vector, which if TRUE applies a further post-processing to the simulated chains (in case these are more than one).

**Value**

A dmbc_fit_list object.

**Author(s)**

Sergio Venturini <sergio.venturini@unicatt.it>
References


See Also

\texttt{bmds} for Bayesian (metric) multidimensional scaling.

\texttt{dmbc_data} for a description of the data format.

\texttt{dmbc_fit_list} for a description of the elements included in the returned object.

Examples

```r
## Not run:
data(simdiss, package = "dmbc")
G <- 3
p <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 20000
nsim <- 10000
seed <- 2301
set.seed(seed)
control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop["z"],
alpha.prop = prm.prop["alpha"], random.start = TRUE, verbose = TRUE,
nchains = 2, thin = 10, store.burnin = TRUE, threads = 2,
parallel = "snow")
sim.dmbc <- dmbc(simdiss, p, G, control)
sim.dmbc <- dmbc(simdiss, p, G, control)
summary(sim.dmbc, include.burnin = FALSE)
library(bayesplot)
library(ggplot2)
color_scheme_set("teal")
plot(sim.dmbc, what = "trace", regex_pars = "eta")
z <- dmbc_get_configuration(sim.dmbc, chain = 1, est = "mean",
labels = 1:16)
summary(z)
color_scheme_set("mix-pink-blue")
graph <- plot(z, size = 2, size_lbl = 3)
graph + panel_bg(fill = "gray90", color = NA)
## End(Not run)
```
**dmbc_check_groups**

*Auxiliary function for checking the grouping results of a fitted DMBC model.*

**Description**

`dmbc_check_groups()` is an auxiliary function for checking whether the cluster membership estimates provided by the individual chains of the fitted model provided agree or not.

**Usage**

```r
dmbc_check_groups(res, est = "mean")
```

**Arguments**

- `res`:
  - An object of class `dmbc_fit_list`.
- `est`:
  - A length-one character vector indicating the estimate type to use.

**Value**

A length-one logical vector, which is equal to TRUE if all simulated chains provide the same cluster membership estimates, and FALSE otherwise.

**Author(s)**

Sergio Venturini <sergio.venturini@unicatt.it>

**References**


**See Also**

- `dmbc_get_configuration()` for a description of the configuration extractor function.
- `dmbc_fit_list` for a description of a fitted DMBC model.

**Examples**

```r
## Not run:
data(simdiss, package = "dmbc")
G <- 3
p <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 2000
nsim <- 1000
```
```r
seed <- 2301

set.seed(seed)

control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop[["z"]],
    alpha.prop = prm.prop[["alpha"]], random.start = TRUE, verbose = TRUE,
    nchains = 2, thin = 10, store.burnin = TRUE, threads = 2,
    parallel = "snow")
sim.dmbc <- dmbc(simdiss, p, G, control)

dmbc_check_groups(sim.dmbc)

## End(Not run)
```

---

**dmbc_config-class**

An S4 class to represent the latent configuration estimate for a DMBC model.

**Description**

An S4 class to represent the latent configuration estimate for a DMBC model.

**Slots**

- **Z.est** An array containing the estimate of the latent configuration for a DMBC model.
- **Z.sd** An array containing the standard deviation of the latent configuration for a DMBC model.
- **cluster** A numeric vector providing the estimated group membership for the \( S \) subjects in the data.
- **est** A length-one character vector providing the estimate type returned in `Z.est`. Possible values are `mean` (posterior mean), `median` (posterior median), `ml` (maximum likelihood) and `map` (maximum-a-posteriori).
- **n** A length-one numeric vector providing the number of objects.
- **p** A length-one numeric vector providing the number of latent dimensions.
- **S** A length-one numeric vector providing the number of subjects.
- **G** A length-one numeric vector providing the number of clusters.
- **family** An object of class `list`; named list with elements representing the parameter estimates corresponding to different values of \( p \) and \( G \).
- **chain** A length-one numeric vector representing the ID of the MCMC chain used to compute the estimates.
- **labels** A character vector for the (optional) strings to use in the plots for labeling the objects.

**References**

Examples

```r
class("dmbc_config")
```

---

**dmbc_control**

**Auxiliary Function for Controlling DMBC Model Fitting**

**Description**

*dmbc_control()* is an auxiliary function as user interface for *dmbc()* fitting. Typically only used when calling the *dmbc()* function. It is used to set parameters that affect the sampling but do not affect the posterior distribution.

*control_dmbc()* is an alias for *dmbc_control()*.

*check_control()* is an auxiliary function that verifies the correctness of the controls provided before a DMBC is fitted with *dmbc()*.

**Usage**

```r
dmbc_control(
  nsim = 5000,
  burnin = 10000,
  thin = 1,
  nchains = 1,
  threads = 1,
  seed = NULL,
  parallel = "no",
  z.prop = 1.5,
  alpha.prop = 0.75,
  random.start = TRUE,
  partition = NULL,
  method = "manhattan",
  procrustes = TRUE,
  relabel = TRUE,
  store.burnin = TRUE,
  verbose = FALSE
)
```

```r
control_dmbc(
  nsim = 5000,
  burnin = 10000,
  thin = 1,
  nchains = 1,
  threads = 1,
  seed = NULL,
  parallel = "no",
  z.prop = 1.5,
)```
$\text{dmbo_control}$

```r
alpha.prop = 0.75,
random.start = TRUE,
partition = NULL,
method = "manhattan",
procrustes = TRUE,
relabel = TRUE,
store.burnin = TRUE,
verbose = FALSE
)
```

check_control(control)

**Arguments**

- **nsim**
  A length-one numeric vector for the number of draws to be taken from the posterior distribution.

- **burnin**
  A length-one numeric vector for the number of initial MCMC iterations (usually to be discarded).

- **thin**
  A length-one numeric vector for the number of iterations between consecutive draws.

- **nchains**
  A length-one numeric vector for the number of parallel chains to run.

- **threads**
  A length-one numeric vector for the number of chains to run. If greater than 1, package `parallel` is used to take advantage of any multiprocessing or distributed computing capabilities that may be available.

- **seed**
  An integer scalar. If supplied, provides the random number seed.

- **parallel**
  A length-one character vector indicating the type of parallel operation to be used (if any). Possible values are `multicore` (which works only on Unix/macOS), `snow`, and `no` (i.e. serial instead of parallel computing).

- **z.prop**
  A length-one numeric vector providing the standard deviation of the proposal distribution for the jump in the individual latent space position.

- **alpha.prop**
  A length-one numeric vector providing the standard deviation of the proposal distribution for the jump in the individual random effect value.

- **random.start**
  A length-one logical vector. If `TRUE` the starting values are drawn randomly, otherwise a user-defined starting partition must be provided through the `partition` argument.

- **partition**
  A length-one numeric vector providing the user-defined starting partition.

- **method**
  A length-one character vector that specifies the distance measure to use in determining the initial partition. Allowed values are those from the `dist()` function.

- **procrustes**
  A length-one logical vector. If `TRUE` the simulated MCMC chains are post-processed through a Procrustes transformation.

- **relabel**
  A length-one logical vector. If `TRUE` the simulated MCMC chains are relabelled to address the label-switching problem.

- **store.burnin**
  A logical scalar. If `TRUE`, the samples from the burnin are also stored and returned.
### dmbc_data-class

An S4 class to represent the data to use in a DMBC model.

#### Description

An S4 class to represent the data to use in a DMBC model.

####Slots

- **diss** A list whose elements are the dissimilarity matrices corresponding to the judgments expressed by the $S$ subjects/raters. These matrices must be defined as a `dist` object.
- **n** A length-one character vector representing the number of objects compared by each subject.
- **S** A length-one numeric vector representing the number of subjects.
- **family** A length-one character vector representing the type of data to analyze. Currently, it accepts only the `"binomial"` value, but future developments will include the possibility to analyze continuous, multinomial and count data.

#### Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>
References

Examples
showClass("dmbc_data")

\begin{verbatim}
dmbc_fit
\end{verbatim}

\textit{Fitter function for DMBC models.}

Description
dmbc_fit() is the main function that estimates a DMBC model.

Usage
dmbc_fit(D, p, G, family, control, prior, start)

Arguments
\begin{itemize}
\item \texttt{D} A list whose elements are the dissimilarity matrices corresponding to the judgments expressed by the $S$ subjects/raters. These matrices must be defined as a \text{dist} object.
\item \texttt{p} A length-one numeric vector indicating the number of dimensions of the latent space.
\item \texttt{G} A length-one numeric vector indicating the number of cluster to partition the $S$ subjects.
\item \texttt{family} A length-one character vector representing the type of data to analyze. Currently, it accepts only the 'binomial' value, but future developments will include the possibility to analyze continuous, multinomial and count data.
\item \texttt{control} A list of control parameters that affect the sampling but do not affect the posterior distribution See \texttt{dmbc_control()} for more details.
\item \texttt{prior} A list containing the prior hyperparameters. See \texttt{dmbc_prior()} for more details.
\item \texttt{start} A named list of starting values for the MCMC algorithm (see \texttt{dmbc_init}).
\end{itemize}

Value
A \texttt{dmbc_fit_list} object.

Author(s)
Sergio Venturini <sergio.venturini@unicatt.it>
References


See Also

dmbc_data for a description of the data format.
dmbc_fit_list for a description of the elements included in the returned object.

Examples

```r
## Not run:
data(simdiss, package = "dmbc")
G <- 3
p <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 20000
nsim <- 10000
seed <- 2301
set.seed(seed)

control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop[["z"]],
                alpha.prop = prm.prop[["alpha"]], random.start = TRUE, verbose = TRUE,
                nchains = 2, thin = 10, store.burnin = TRUE, threads = 2,
                parallel = "snow")
sim.dmbc <- dmbc(simdiss, p, G, control)

summary(sim.dmbc, include.burnin = FALSE)

library(bayesplot)
library(ggplot2)
color_scheme_set("teal")
plot(sim.dmbc, what = "trace", regex_pars = "eta")

z <- dmbc_get_configuration(sim.dmbc, chain = 1, est = "mean",
                            labels = 1:16)
summary(z)
color_scheme_set("mix-pink-blue")
graph <- plot(z, size = 2, size_lbl = 3)
graph + panel_bg(fill = "gray90", color = NA)

## End(Not run)
```
An S4 class to represent the results of fitting DMBC model.

Description

An S4 class to represent the results of fitting DMBC model using a single Markov Chain Monte Carlo chain.

Slots

- `z.chain` An object of class `array`; posterior draws from the MCMC algorithm for the (untransformed) latent configuration $Z$.
- `z.chain.p` An object of class `array`; posterior draws from the MCMC algorithm for the (Procrustes-transformed) latent configuration $Z$.
- `alpha.chain` An object of class `matrix`; posterior draws from the MCMC algorithm for the $\alpha$ parameters.
- `eta.chain` An object of class `matrix`; posterior draws from the MCMC algorithm for the $\eta$ parameters.
- `sigma2.chain` An object of class `matrix`; posterior draws from the MCMC algorithm for the $\sigma^2$ parameters.
- `lambda.chain` An object of class `matrix`; posterior draws from the MCMC algorithm for the $\lambda$ parameters.
- `prob.chain` An object of class `array`; posterior draws from the MCMC algorithm for the cluster membership probabilities.
- `x.ind.chain` An object of class `array`; posterior draws from the MCMC algorithm for the cluster membership indicators.
- `x.chain` An object of class `matrix`; posterior draws from the MCMC algorithm for the cluster membership labels.
- `accept` An object of class `matrix`; final acceptance rates for the MCMC algorithm.
- `diss` An object of class `list`; list of observed dissimilarity matrices.
- `dens` An object of class `list`; list of log-likelihood, log-prior and log-posterior values at each iteration of the MCMC simulation.
- `control` An object of class `list`; list of the control parameters (number of burnin and sample iterations, number of MCMC chains, etc.). See `dmbc_control()` for more information.
- `prior` An object of class `list`; list of the prior hyperparameters. See `dmbc_prior()` for more information.
- `dim` An object of class `list`; list of dimensions for the estimated model, i.e. number of objects ($n$), number of latent dimensions ($p$), number of clusters ($G$), and number of subjects ($S$).
- `model` An object of class `dmbc_model`.


References


Examples

\texttt{showClass("dmbc_fit")}

---

\texttt{dmbc_fit_list-class}  \textit{An S4 class to represent the results of fitting DMBC model.}

Description

An S4 class to represent the results of fitting DMBC model using multiple Markov Chain Monte Carlo chains.

Slots

\texttt{results} An object of class \texttt{list}; list of \texttt{dmbc_fit} objects corresponding to the parallel MCMC chains simulated during the estimation.

References


See Also

\texttt{dmbc_fit} for more details on the components of each element of the list.

Examples

\texttt{showClass("dmbc_fit_list")}
**dmbc_fit_list_to_list**  
Conversion of an dmbc_fit_list object to a list.

### Description

dmbc_fit_list_to_list converts an object of class dmbc_fit_list to a list of arrays including all the parameter chains. It is intended for internal use mainly.

### Usage

dmbc_fit_list_to_list(res, include.burnin = FALSE, verbose = TRUE)

### Arguments

- **res**  
  An object of type dmbc_fit_list.

- **include.burnin**  
  A logical scalar. If TRUE the burnin iterations (if available) are not removed.

- **verbose**  
  A logical scalar. If TRUE prints additional warnings during the conversion.

### Value

An object of type mcmc.list.

### Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

### See Also

dmbc() for for fitting a DMBC model; dmbc_fit_list-class.

### Examples

```r
## Not run:
data(simdiss, package = "dmbc")

G <- 3
p <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 2000
nsim <- 1000
seed <- 2301

set.seed(seed)

control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop["z"],
                alpha.prop = prm.prop["alpha"], nchains = 2, verbose = TRUE)
sim.dmbc <- dmbc(simdiss, p, G, control)
sim.list <- dmbc_fit_list_to_list(sim.dmbc, TRUE)
```
library(bayesplot)
mcmc_trace(sim.list, regex_pars = "lambda")

## End(Not run)

dmbc_fit_list_to_mcmc.list

Conversion of an dmbc_fit_list object to an object of class mcmc.list.

Description

dmbc_fit_list_to_mcmc.list converts an object of class dmbc_fit_list to an object with class mcmc.list.

Usage

dmbc_fit_list_to_mcmc.list(res, include.burnin = FALSE, verbose = TRUE)

Arguments

res
An object of type dmbc_fit_list.

include.burnin
A logical scalar. If TRUE the burnin iterations (if available) are not removed.

verbose
A logical scalar. If TRUE prints additional warnings during the conversion.

Value

An object of type mcmc.list.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

See Also

dmbc() for fitting a DMBC model; dmbc_fit_list-class; mcmc.list.

Examples

## Not run:
data(simdiss, package = "dmbc")

G <- 3
p <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 2000
nsim <- 1000
seed <- 2301
set.seed(seed)

control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop["z"],
alpha.prop = prm.prop["alpha"], nchains = 2, verbose = TRUE)
sim.dmbc <- dmbc(simdiss, p, G, control)
sim.mcmc <- dmbc_fit_list_to_mcmc.list(sim.dmbc, TRUE)
plot(sim.mcmc)

## End(Not run)

dmbc_fit_to_mcmc

Conversion of an dmbc_fit object to an object of class mcmc.

Description
dmbc_fit_to_mcmc converts an object of class dmbc_fit to an object with class mcmc.

Usage
dmbc_fit_to_mcmc(res, include.burnin = FALSE, verbose = TRUE)

Arguments
res An object of type dmbc_fit.
include.burnin A logical scalar. If TRUE the burnin iterations (if available) are not removed.
verbose A logical scalar. If TRUE prints additional warnings during the conversion.

Value
An object of type mcmc.

Author(s)
Sergio Venturini <sergio.venturini@unicatt.it>

See Also
dmbc() for for fitting a DMBC model; dmbc_fit-class; mcmc.

Examples
## Not run:
data(simdiss, package = "dmbc")

G <- 3
p <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 2000
nsim <- 1000
seed <- 2301

set.seed(seed)

control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop["z"],
alpha.prop = prm.prop["alpha"], verbose = TRUE)
sim.dmbc <- dmbc(simdiss, p, G, control)
sim.mcmc <- dmbc_fit_to_mcmc(sim.dmbc$results[[1]], TRUE)
plot(sim.mcmc)

## End(Not run)

dmbc_get_configuration

Extractor function for a fitted DMBC model.

Description

dmbc_get_configuration() is an extractor function for extracting the latent configuration estimates of a fitted DMBC model.

Usage

dmbc_get_configuration(res, chain = 1, est = "mean", labels = character(0))

Arguments

res
An object of class dmbc_fit_list.

chain
A length-one numeric vector indicating the MCMC chain number to use.

est
A length-one character vector indicating the estimate type to use.

labels
An optional character vector with the object labels.

Value

A dmbc_config object.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

References

Extractor function for a fitted DMBC model.

Description

dmbc_get_map() is an extractor function for extracting the maximum-a-posterior estimates of the parameters for a fitted DMBC model.

Usage

dmbc_get_map(res, chain = 1)
Arguments

res
An object of class dmbc_fit_list.

chain
A length-one numeric vector indicating the MCMC chain number to use.

Value

A named list with the following elements:

z: array of latent coordinates posterior mean estimates
alpha: numeric vector of alpha posterior mean estimates
eta: numeric vector of eta posterior mean estimates
sigma2: numeric vector of sigma2 posterior mean estimates
lambda: numeric vector of lambda posterior mean estimates
prob: numeric matrix of probability posterior mean estimates
cluster: numeric vector of cluster membership posterior mean estimates
logpost: length-one numeric vector of the maximum log-posterior value
chain: length-one numeric vector of the MCMC chain number used

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

References


See Also
dmbc_data for a description of the data format.
dmbc_fit_list for a description of the elements included in the returned object.

Examples

## Not run:
data(simdiss, package = "dmbc")

G <- 3
p <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 2000
nsim <- 1000
seed <- 2301

set.seed(seed)
```r
control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop["z"],
                alpha.prop = prm.prop["alpha"], random.start = TRUE, verbose = TRUE,
                nchains = 2, thin = 10, store.burnin = TRUE, threads = 2,
                parallel = "snow")
sim.dmbc <- dmbc(simdiss, p, G, control)

dmbc_get_map(sim.dmbc, chain = 1)

## End(Not run)
```

**Description**

`dmbc_get_ml()` is an extractor function for extracting the maximum likelihood estimates of the parameters for a fitted DMBC model.

**Usage**

```r
dmbc_get_ml(res, chain = 1)
```

**Arguments**

- `res`: An object of class `dmbc_fit_list`.
- `chain`: A length-one numeric vector indicating the MCMC chain number to use.

**Value**

A named list with the following elements:

- `z`: array of latent coordinates posterior mean estimates
- `alpha`: numeric vector of alpha posterior mean estimates
- `eta`: numeric vector of eta posterior mean estimates
- `sigma2`: numeric vector of sigma2 posterior mean estimates
- `lambda`: numeric vector of lambda posterior mean estimates
- `prob`: numeric matrix of probability posterior mean estimates
- `cluster`: numeric vector of cluster membership posterior mean estimates
- `loglik`: length-one numeric vector of the maximum log-likelihood value
- `chain`: length-one numeric vector of the MCMC chain number used

**Author(s)**

Sergio Venturini `<sergio.venturini@unicatt.it>`
References

See Also
\texttt{dmbc_data} for a description of the data format.
\texttt{dmbc_fit_list} for a description of the elements included in the returned object.

Examples
## Not run:
data(simdiss, package = "dmbc")

G <- 3
p <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 2000
nsim <- 1000
seed <- 2301

set.seed(seed)

control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop["z"],
               alpha.prop = prm.prop["alpha"], random.start = TRUE, verbose = TRUE,
               nchains = 2, thin = 10, store.burnin = TRUE, threads = 2,
               parallel = "snow")
sim.dmbc <- dmbc(simdiss, p, G, control)

dmbc_get_ml(sim.dmbc, chain = 1)

## End(Not run)

\begin{verbatim}
 dmbc_get_postmean \hspace{1cm} Extractor function for a fitted DMBC model.
\end{verbatim}

Description
\texttt{dmbc_get_postmean()} is an extractor function for extracting the posterior mean estimates of the parameters for a fitted DMBC model.

Usage
\texttt{dmbc_get_postmean(res, chain = 1)}
Arguments

res
An object of class dmbc_fit_list.

chain
A length-one numeric vector indicating the MCMC chain number to use.

Value

A named list with the following elements:

z: array of latent coordinates posterior mean estimates
alpha: numeric vector of alpha posterior mean estimates
eta: numeric vector of eta posterior mean estimates
sigma2: numeric vector of sigma2 posterior mean estimates
lambda: numeric vector of lambda posterior mean estimates
prob: numeric matrix of probability posterior mean estimates
cluster: numeric vector of cluster membership posterior mean estimates
chain: length-one numeric vector of the MCMC chain number used

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

References


See Also

dmbc_data for a description of the data format.
dmbc_fit_list for a description of the elements included in the returned object.

Examples

## Not run:
data(simdiss, package = "dmbc")

G <- 3
p <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 2000
nsim <- 1000
seed <- 2301

set.seed(seed)

control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop[["z"]],
alpha.prop = prm.prop[["alpha"]], random.start = TRUE, verbose = TRUE,
nchains = 2, thin = 10, store.burnin = TRUE, threads = 2,
parallel = "snow")
sim.dmbc <- dmbc(simdiss, p, G, control)

dmbc_get_postmedian(sim.dmbc, chain = 1)

## End(Not run)

dmbc_get_postmedian Extractor function for a fitted DMBC model.

Description

dmbc_get_postmedian() is an extractor function for extracting the posterior median estimates of
the parameters for a fitted DMBC model.

Usage

dmbc_get_postmedian(res, chain = 1)

Arguments

res An object of class dmbc_fit_list.
chain A length-one numeric vector indicating the MCMC chain number to use.

Value

A named list with the following elements:

z: array of latent coordinates posterior median estimates
alpha: numeric vector of alpha posterior median estimates
eta: numeric vector of eta posterior median estimates
sigma2: numeric vector of sigma2 posterior median estimates
lambda: numeric vector of lambda posterior median estimates
prob: numeric matrix of probability posterior median estimates
cluster: numeric vector of cluster membership posterior median estimates
chain: length-one numeric vector of the MCMC chain number used

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

References

Venturini, S., Piccarreta, R. (2021), "A Bayesian Approach for Model-Based Clustering of Several
Binary Dissimilarity Matrices: the dmbc Package in R", Journal of Statistical Software, 100, 16,
1–35, <10.18637/jss.v100.i16>.
See Also

- `dmbc_data` for a description of the data format.
- `dmbc_fit_list` for a description of the elements included in the returned object.

Examples

```r
## Not run:
data(simdiss, package = "dmbc")

G <- 3
p <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 2000
nsim <- 1000
seed <- 2301
set.seed(seed)
control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop[["z"]],
                alpha.prop = prm.prop[["alpha"]], random.start = TRUE, verbose = TRUE,
                nchains = 2, thin = 10, store.burnin = TRUE, threads = 2,
                parallel = "snow")
sim.dmbc <- dmbc(simdiss, p, G, control)
dmbc_get_postmedian(sim.dmbc, chain = 1)
## End(Not run)
```

---

### `dmbc_IC`

*Model selection of DMBC models.*

**Description**

`dmbc_IC()` is the main function for simultaneously selecting the optimal latent space dimension ($p$) and number of clusters ($G$) for a DMBC analysis.

**Usage**

```r
dmbc_IC(
data,  
pmax = 3,  
Gmax = 5,  
control = dmbc_control(),  
prior = NULL,  
est = "mean"
)
```
Arguments

data  An object of class dmbc_data containing the data to analyze.
pmax  A length-one numeric vector indicating the maximum number of dimensions of the latent space to consider.
Gmax  A length-one numeric vector indicating the maximum number of cluster to consider.
control  A list of control parameters that affect the sampling but do not affect the posterior distribution. See dmbc_control() for more details.
prior  A list containing the prior hyperparameters. See dmbc_prior() for more details.
est  A length-one character vector indicating the estimate type to use. Possible values are mean, median, ml and map.

Value

A dmbc_ic object.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

References


See Also

dmbc() for fitting a DMBC model.
dmbc_ic for a description of the elements included in the returned object.

Examples

```r
## Not run:
data(simdiss, package = "dmbc")
pmax <- 2
Gmax <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 2000
nsim <- 1000
seed <- 1809
set.seed(seed)

control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop[["z"]],
                alpha.prop = prm.prop[["alpha"]], random.start = TRUE, verbose = TRUE,
                thin = 10, store.burnin = TRUE)
```
```r
dmbc_ic <- dmbc_IC(data = simdiss, pmax = pmax, Gmax = Gmax, control = control, est = "mean")
pmax <- pmax + 1
Gmax <- Gmax + 2
new.ic <- update(sim.ic, pmax = pmax, Gmax = Gmax)
new.ic

# plot the results
library(bayesplot)
library(ggplot2)
color_scheme_set("mix-yellow-blue")
p <- plot(new.ic, size = c(4, 1.5))
p + panel_bg(fill = "gray90", color = NA)
```

---

**dmbc_ic-class**

An S4 class to represent the comparison of a set of DMBC models.

---

**Description**

An S4 class to represent the comparison of a set of DMBC models through the dissimilarity model-based clustering information criterion (DCIC).

**Slots**

- `logprior` An object of class `matrix` providing the log-prior values corresponding to different values of `p` and `G`.
- `logmlik` An object of class `matrix` providing the marginal log-likelihood values corresponding to different values of `p` and `G`.
- `logcorrfact` An object of class `matrix` providing the logarithm of the correction factors corresponding to different values of `p` and `G`.
- `DCIC` An object of class `matrix` providing the values of the DCIC index corresponding to different values of `p` and `G`.
- `post.est` An object of class `list`; named list with elements representing the parameter estimates corresponding to different values of `p` and `G`.
- `est` A length-one character vector representing the estimate type used in computing the DCIC index. Possible values are `mean`, `median`, `ml` and `map`. See `dmbc_ic()` for more details about these values.
- `res_last_p` An object of class `list`; list of `dmbc_fit_list` objects with the results of fitting the DMBC models corresponding to the last value of `p`. This is needed in case of an update of the DCIC calculations using additional `p` and/or `G` values.
References


Examples

showClass("dmbc_ic")

---

**dmbc_init**

*Function to compute the starting values before fitting a DMBC models.*

**Description**

`dmbc_init()` is the main function that estimates a DMBC model.

**Usage**

```r
dmbc_init(D, p, G, family, random.start, method, partition)
```

**Arguments**

- **D**: A list whose elements are the dissimilarity matrices corresponding to the judgments expressed by the S subjects/raters. These matrices must be defined as a `dist` object.
- **p**: A length-one numeric vector indicating the number of dimensions of the latent space.
- **G**: A length-one numeric vector indicating the number of cluster to partition the S subjects.
- **family**: A length-one character vector representing the type of data to analyze. Currently, it accepts only the 'binomial' value, but future developments will include the possibility to analyze continuous, multinomial and count data.
- **random.start**: A length-one logical vector. If `TRUE` the starting values are drawn randomly, otherwise.
- **method**: A length-one character vector specifying the distance measure to use in determining the initial partition. Allowed values are those from the `dist()` function.
- **partition**: A length-one numeric vector providing the user-defined starting partition.
Value

A named list with the following items:

- **z**: array of latent coordinates starting values
- **x**: numeric vector of initial cluster memberships
- **ng**: numeric vector of initial cluster sizes
- **alpha**: numeric vector of alpha starting values
- **eta**: numeric vector of eta starting values
- **sigma2**: numeric vector of sigma2 starting values
- **lambda**: numeric vector of lambda starting values

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

References


See Also

dmbc() for fitting a DMBC model.

Examples

data(simdiss, package = "dmbc")
dmbc_init(simdiss@diss, p = 2, G = 3, family = "binomial", random.start = TRUE)

```

| dmbc_logLik | Log-likelihood for DMBC models. |

Description

dmbc_logLik() computes the log-likelihood value for a DMBC model.

Usage

dmbc_logLik(D, Z, alpha, lambda, x)
Arguments

D
A list whose elements are the dissimilarity matrices corresponding to the judgments expressed by the S subjects/raters. These matrices must be defined as a dist object.

Z
A numeric matrix containing the latent configuration.

alpha
A numeric vector containing the alpha values.

lambda
A numeric vector containing the alpha lambda.

x
A numeric vector containing the cluster indicator values.

Value

A length-one numeric vector of the log-likelihood value.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

References


See Also

dmbc().
**Value**

A length-one numeric vector of the log-likelihood value.

**Author(s)**

Sergio Venturini <sergio.venturini@unicatt.it>

**References**


**See Also**

\texttt{dmbc}().

---

\texttt{dmbc_match_groups}  

\textit{Auxiliary function for realigning the grouping of a fitted DMBC model.}

**Description**

\texttt{dmbc_match_groups}() is an auxiliary function for realigning the cluster membership estimates provided by the individual chains of the fitted model if they do not agree.

**Usage**

\texttt{dmbc_match_groups(res, est = "mean", ref = 1)}

**Arguments**

- \texttt{res}  
  An object of class \texttt{dmbc_fit_list}.
- \texttt{est}  
  A length-one character vector indicating the estimate type to use.
- \texttt{ref}  
  A length-one numeric vector indicating the chain number to use as the reference.

**Value**

An object of class \texttt{dmbc_fit_list}.

**Author(s)**

Sergio Venturini <sergio.venturini@unicatt.it>

**References**

See Also

dmbc_check_groups() for checking the consistency of the cluster memberships across chains for a fitted DMBC model.
dmbc_get_configuration() for a description of the configuration extractor function.
dmbc_fit_list for a description of a fitted DMBC model.

Examples

```r
## Not run:
data(simdiss, package = "dmbc")

G <- 5
p <- 3
prm.prop <- list(z = 4, alpha = 2)
burnin <- 2000
nsim <- 1000
seed <- 2301

set.seed(seed)

control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop["z"],
alpha.prop = prm.prop["alpha"], random.start = TRUE, verbose = TRUE,
nchains = 6, store.burnin = TRUE, threads = 2, parallel = "snow")
sim.dmbc <- dmbc(simdiss, p, G, control)

sim.dmbc_new <- dmbc_match_groups(sim.dmbc)

## End(Not run)
```

dmbc_mcmc

`dmbc_mcmc` Internal functions for MCMC simulation.

Description

For internal use only.

Usage

`dmbc_mcmc`

```r
dmbc_mcmc(
  raiD,
  raix,
  raing,
  radalpha,
  rn,
  rp,
  rG,
  rS,
)```

Arguments

raiD  internal SEXP data structure
raix  internal SEXP data structure
raing internal SEXP data structure
radalpha internal SEXP data structure
rn    internal SEXP data structure
rp    internal SEXP data structure
rG    internal SEXP data structure
rS    internal SEXP data structure
### Slots

- **p** A length-one character vector representing the number of dimensions of the latent space to use in the MDS analysis.
- **G** A length-one numeric vector representing the number of clusters to partition the subjects into.
- **family** A length-one character vector representing the type of data to analyze. Currently, it accepts only the 'binomial' value, but future developments will include the possibility to analyze continuous, multinomial and count data.

---

**dmbc_model-class**

*An S4 class to represent a DMBC model.*

**Description**

An S4 class to represent a DMBC model.
References

Examples
showClass("dmbc_model")

dmbc_prior

\begin{description}
\item [\texttt{Description}] \texttt{dmbc_prior}() is an auxiliary function as user interface for \texttt{dmbc()} fitting. Typically only used when calling the \texttt{dmbc()} function. It is used to set prior hyperparameters.
\item [\texttt{prior_dmbc}() is an alias for \texttt{dmbc_prior}().]
\item [\texttt{check_prior}() is an auxiliary function that verifies the correctness of the prior hyperparameters provided before a DMBC is fitted with \texttt{dmbc}().]
\item [\texttt{update_prior}() is an auxiliary function to modify a set of prior choices using a new value of \texttt{p} and \texttt{G}. It is intended for internal use mainly in the \texttt{dmbc_ic}() function.]
\end{description}

\textbf{Usage}
\begin{verbatim}
dmbc_prior(
  eta = list(a = rep(1.5, .dmbcEnv$current_G), b = rep(0.5, .dmbcEnv$current_G)),
  sigma2 = list(a = 0.1, b = 0.1),
  lambda = rep(1, .dmbcEnv$current_G)
)
prior_dmbc(
  eta = list(a = rep(1.5, .dmbcEnv$current_G), b = rep(0.5, .dmbcEnv$current_G)),
  sigma2 = list(a = 0.1, b = 0.1),
  lambda = rep(1, .dmbcEnv$current_G)
)
check_prior(prior)
update_prior(prior, p, G)
\end{verbatim}

\textbf{Arguments}
\begin{itemize}
\item [\texttt{eta}] A named list containing the hyperparameters for the prior distribution of the $\eta_1, \ldots, \eta_G$ parameters. It should contain two numeric vectors, namely \texttt{a} and \texttt{b}.
\item [\texttt{sigma2}] A list containing the hyperparameters for the prior distribution of the $\sigma_2$ parameters. It should contain two numeric values, namely \texttt{a} and \texttt{b}.
\item [\texttt{lambda}] A numeric vector containing the hyperparameters for the prior distribution of the $\lambda$ parameters.
\end{itemize}
initialize, dmbc_config-method

Create an instance of the dmbc_config class using new/initilize.

Description

Create an instance of the dmbc_config class using new/initilize.

sigma2  A named list containing the hyperparameters for the prior distributions of the $\sigma_1^2, \ldots, \sigma_G^2$ parameters. It should contain two numeric scalars, namely $a$ and $b$.

lambda  A list containing the hyperparameters for the prior distribution of the $\lambda_1, \ldots, \lambda_G$ parameters. It should contain a single numeric vector.

prior  A named list of prior hyperparameters.

p  A length-one numeric vector indicating the number of dimensions of the latent space.

G  A length-one numeric vector indicating the number of cluster to partition the $S$ subjects.

Value

A list with the prior hyperparameters as components.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

See Also

dmbc()

Examples

## Not run:
data(simdiss, package = "dmbc")
# Shorter run than default.
sim.fit <- dmbc(simdiss,
  control = dmbc_control(burnin = 1000, nsim = 2000, thin = 1, verbose = TRUE),
  prior = dmbc_prior(sigma2 = list(a = 1, b = 4)))
## End(Not run)
Usage

```r
## S4 method for signature 'dmbc_config'
initialize(
  .Object,
  Z.est = array(),
  Z.sd = array(),
  cluster = numeric(),
  est = character(),
  n = numeric(),
  S = numeric(),
  p = numeric(),
  G = numeric(),
  family = character(),
  chain = numeric(),
  labels = character()
)
```

Arguments

- `.Object` Prototype object from the class `dmbc_config`.
- `Z.est` An array containing the estimate of the latent configuration for a DMBC model.
- `Z.sd` An array containing the standard deviation of the latent configuration for a DMBC model.
- `cluster` A numeric vector providing the estimated group membership for the `S` subjects in the data.
- `est` A length-one character vector providing the estimate type returned in `Z.est`. Possible values are `mean` (posterior mean), `median` (posterior median), `ml` (maximum likelihood) and `map` (maximum-a-posteriori).
- `n` A length-one numeric vector providing the number of objects.
- `S` A length-one numeric vector providing the number of subjects.
- `p` A length-one numeric vector providing the number of latent dimensions.
- `G` A length-one numeric vector providing the number of clusters.
- `family` An object of class `list`; named list with elements representing the parameter estimates corresponding to different values of `p` and `G`.
- `chain` A length-one numeric vector representing the ID of the MCMC chain used to compute the estimates.
- `labels` A character vector for the (optional) strings to use in the plots for labeling the objects.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>
Create an instance of the `dmbc_data` class using `new`/`initialize`.

### Usage

```r
## S4 method for signature 'dmbc_data'
initialize(
  .Object,
  diss = list(),
  n = numeric(),
  S = numeric(),
  family = character()
)
```

### Arguments

- `.Object` Prototype object from the class `dmbc_data`.
- `diss` A list whose elements are the dissimilarity matrices corresponding to the judgments expressed by the `S` subjects/raters. These matrices must be defined as a `dist` object.
- `n` A length-one character vector representing the number of objects compared by each subject.
- `S` A length-one numeric vector representing the number of subjects.
- `family` A length-one character vector representing the type of data to analyze. Currently, it accepts only the 'binomial' value, but future developments will include the possibility to analyze continuous, multinomial and count data.

### Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

Create an instance of the `dmbc_fit` class using `new`/`initialize`.

### Usage

```r
## S4 method for signature 'dmbc_fit'
initialize(
  .Object,
  diss = list(),
  n = numeric(),
  S = numeric(),
  family = character()
)
```

### Arguments

- `.Object` Prototype object from the class `dmbc_fit`.
- `diss` A list whose elements are the dissimilarity matrices corresponding to the judgments expressed by the `S` subjects/raters. These matrices must be defined as a `dist` object.
- `n` A length-one character vector representing the number of objects compared by each subject.
- `S` A length-one numeric vector representing the number of subjects.
- `family` A length-one character vector representing the type of data to analyze. Currently, it accepts only the 'binomial' value, but future developments will include the possibility to analyze continuous, multinomial and count data.

### Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>
initialize.dmhc_fit-method

Usage

```r
## S4 method for signature 'dmhc_fit'
initialize(  .Object,  z.chain = array(),  z.chain.p = array(),  alpha.chain = matrix(),  eta.chain = matrix(),  sigma2.chain = matrix(),  lambda.chain = matrix(),  prob.chain = array(),  x.ind.chain = array(),  x.chain = matrix(),  accept = matrix(),  diss = list(),  dens = list(),  control = list(),  prior = list(),  dim = list(),  model = NA)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.Object</td>
<td>Prototype object from the class <code>dmhc_fit</code>.</td>
</tr>
<tr>
<td>z.chain</td>
<td>An object of class <code>array</code>; posterior draws from the MCMC algorithm for the (untransformed) latent configuration $Z$.</td>
</tr>
<tr>
<td>z.chain.p</td>
<td>An object of class <code>array</code>; posterior draws from the MCMC algorithm for the (Procrustes-transformed) latent configuration $\tilde{Z}$.</td>
</tr>
<tr>
<td>alpha.chain</td>
<td>An object of class <code>matrix</code>; posterior draws from the MCMC algorithm for the $\alpha$ parameters.</td>
</tr>
<tr>
<td>eta.chain</td>
<td>An object of class <code>matrix</code>; posterior draws from the MCMC algorithm for the $\eta$ parameters.</td>
</tr>
<tr>
<td>sigma2.chain</td>
<td>An object of class <code>matrix</code>; posterior draws from the MCMC algorithm for the $\sigma^2$ parameters.</td>
</tr>
<tr>
<td>lambda.chain</td>
<td>An object of class <code>matrix</code>; posterior draws from the MCMC algorithm for the $\lambda$ parameters.</td>
</tr>
<tr>
<td>prob.chain</td>
<td>An object of class <code>array</code>; posterior draws from the MCMC algorithm for the cluster membership probabilities.</td>
</tr>
<tr>
<td>x.ind.chain</td>
<td>An object of class <code>array</code>; posterior draws from the MCMC algorithm for the cluster membership indicators.</td>
</tr>
<tr>
<td>x.chain</td>
<td>An object of class <code>matrix</code>; posterior draws from the MCMC algorithm for the cluster membership labels.</td>
</tr>
<tr>
<td>accept</td>
<td>An object of class <code>matrix</code>; final acceptance rates for the MCMC algorithm.</td>
</tr>
<tr>
<td>diss</td>
<td>An object of class <code>list</code>; list of observed dissimilarity matrices.</td>
</tr>
</tbody>
</table>
### initialize, dmbc_fit_list-method

Create an instance of the `dmbc_fit_list` class using `new/initialize`.

#### Description

Create an instance of the `dmbc_fit_list` class using `new/initialize`.

#### Usage

```r
## S4 method for signature 'dmbc_fit_list'
initialize(.Object, results = list())
```

#### Arguments

- `.Object`: Prototype object from the class `dmbc_fit_list`.
- `results`: A list whose elements are the `dmbc_fit` objects for each simulated chain.

#### Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>
Create an instance of the \texttt{dmbc_ic} class using \texttt{new}/\texttt{initialize}.

## S4 method for signature 'dmbc_ic'

\begin{verbatim}
initialize(
  .Object,
  logprior = matrix(),
  logmlik = matrix(),
  logcorrfact = matrix(),
  DCIC = matrix(),
  post.est = list(),
  est = character(),
  res_last_p = list()
)
\end{verbatim}

### Arguments

- \texttt{.Object} Prototype object from the class \texttt{dmbc_ic}.
- \texttt{logprior} An object of class \texttt{matrix} providing the log-prior values corresponding to different values of \(p\) and \(G\).
- \texttt{logmlik} An object of class \texttt{matrix} providing the marginal log-likelihood values corresponding to different values of \(p\) and \(G\).
- \texttt{logcorrfact} An object of class \texttt{matrix} providing the logarithm of the correction factors corresponding to different values of \(p\) and \(G\).
- \texttt{DCIC} An object of class \texttt{matrix} providing the values of the DCIC index corresponding to different values of \(p\) and \(G\).
- \texttt{post.est} An object of class \texttt{list}; named list with elements representing the parameter estimates corresponding to different values of \(p\) and \(G\).
- \texttt{est} A length-one character vector representing the estimate type used in computing the DCIC index. Possible values are \texttt{mean}, \texttt{median}, \texttt{ml} and \texttt{map}. See \texttt{dmbc_ic()} for more details about these values.
- \texttt{res_last_p} An object of class \texttt{list}; list of \texttt{dmbc_fit_list} objects with the results of fitting the DMBC models corresponding to the last value of \(p\). This is needed in case of an update of the DCIC calculations using additional \(p\) and/or \(G\) values.

### Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>
initialize, dmbc_model-method

Create an instance of the dmbc_model class using new/initialize.

Description

Create an instance of the dmbc_model class using new/initialize.

Usage

```r
## S4 method for signature 'dmbc_model'
initialize(.Object, p = numeric(), G = numeric(), family = character())
```

Arguments

- `.Object`: Prototype object from the class `dmbc_model`.
- `p`: A length-one character vector representing the number of dimensions of the latent space to use in the MDS analysis.
- `G`: A length-one numeric vector representing the number of clusters to partition the subjects into.
- `family`: A length-one character vector representing the type of data to analyze. Currently, it accepts only the 'binomial' value, but future developments will include the possibility to analyze continuous, multinomial and count data.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

kinship

List of binary dissimilarity matrices among 15 kinship terms.

Description

Rosenberg and Kim (1975) designed an experiment to analyze the perceived similarities of 15 kinship terms. Here, we consider the data relative to 85 females made available in Rosenberg (1982). Each subject was asked to group the kinship terms according to the perceived similarity. Thus, \( S = 85 \) binary dissimilarity matrices are available whose elements (0 or 1) indicate whether or not two kinship terms were grouped together by each individual.

Usage

data(kinship)
**Format**

A `dmbc_data` object whose `diss` element is a list of 85 binary dissimilarity matrices. Each matrix is defined as a `dist` object measuring whether each pair of the 15 kinship terms is judged as similar (1) or not (0).

The `dist` objects have rows and columns that are named as follows:

- **GrF** grandfather
- **GrM** grandmother
- **GrD** granddaughter
- **GrS** grandson
- **Bro** brother
- **Sis** sister
- **Fat** father
- **Mot** mother
- **Dau** daughter
- **Son** son
- **Nep** nephew
- **Nie** niece
- **Cou** cousin
- **Aun** aunt
- **Unc** uncle

**References**


**Examples**

```r
data(kinship)
library(bayesplot)
cols <- color_scheme_set("mix-red-blue")
plot(kinship, colors = unlist(cols)[c(1, 6)], font = 1, cex.font = 0.75)
```
Information criterion for Bayesian multidimensional scaling (BMDS).

Description

`mdsic` computes the information criterion for a set of Bayesian multidimensional scaling (BMDS) solutions using the approach in Oh & Raftery (2001).

Usage

```r
mdsic(x_star, rmin_ssr, n, min_p = 1, max_p = 6)
```

Arguments

- `x_star` An array containing the latent configurations estimated using `bmds`.
- `rmin_ssr` A numeric vector providing the ratios of SSR for the latent dimensions requested.
- `n` A length-one numeric vector providing the number of objects.
- `min_p` A length-one numeric vector providing the minimum value of the latent space dimension to use.
- `max_p` A length-one numeric vector providing the maximum value of the latent space dimension to use.

Value

A list with the following elements:

- `mdsic` A numeric vector with the values of MDSIC index.
- `bic` A numeric vector with the values of the BIC index.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

References


See Also

- `bmds` for Bayesian (metric) multidimensional scaling and `comp_ssr` for the computation of SSR.
Examples

```r
## Not run:
# Road distances (in km) between 21 cities in Europe
data(eurodist, package = "datasets")

min_p <- 1
max_p <- 10
burnin <- 200
nsim <- 1000
totiter <- burnin + nsim

eurodist.mds <- cmdscale(eurodist, max_p)
eurodist.bmds <- bmds(eurodist, min_p, max_p, burnin, nsim)

plot((min_p:max_p), eurodist.bmds$mdsIC$mdsic, type = "b",
     main = "MDS Information Criterion", xlab = "p", ylab = "MDSIC")
MDSICmin <- which.min(eurodist.bmds$mdsIC$mdsic)
points((min_p:max_p)[MDSICmin], eurodist.bmds$mdsIC$mdsic[MDSICmin],
     col = "red", pch = 10, cex = 1.75, lwd = 1.5)

## End(Not run)
```

Description

Provide a graphical summary of a `dmbc_config` class instance.

Usage

```r
# S4 method for signature 'dmbc_config,ANY'
plot(
  x,
  size = NULL,
  size_lbl = NULL,
  nudge_x = 0,
  nudge_y = 0,
  label_objects = TRUE,
  ...
)
```

Arguments

- `x` An object of class `dmbc_config`.
- `size` A length-two numeric vector providing the optional sizes of points and lines in the plot.
provide a graphical summary of a dmbc_data class instance.

## S4 method for signature 'dmbc_data,ANY'

plot(x, colors = c("white", "black"), font = NA, cex.font = NA, ...)

Arguments

- **x**
  - An object of class `dmbc_data`.
- **colors**
  - A character vector providing the colors to use in the plot.
- **font**
  - A length-one numeric vector for the font to use for text. Can be a vector. NA values (the default) mean use `par("font")`.
- **cex.font**
  - A length-one numeric vector for the character expansion factor. NULL and NA are equivalent to 1.0. This is an absolute measure, not scaled by `par("cex")` or by setting `par("mfrac")` or `par("mfcol")`. Can be a vector.
  - Further arguments to pass on (currently ignored).

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

Examples

data(simdiss)
library(bayesplot)
cols <- color_scheme_set("brightblue")
plot(simdiss, colors = unlist(cols)[c(1, 6)], font = 1, cex.font = 0.75)
plot.dmbc_fit,ANY-method

Provide a graphical summary of a dmbc_fit class instance.

Description

Provide a graphical summary of a dmbc_fit class instance.

Usage

```r
## S4 method for signature 'dmbc_fit,ANY'
plot(
  x,
  what = "trace",
  pars = character(),
  regex_pars = "lambda",
  include.burnin = FALSE,
  combo = NULL,
  ...
)
```

Arguments

- **x** An object of class `dmbc_fit`.
- **what** A length-one character vector providing the plot type to produce. Admissible values are those provided by the `bayesplot` package, that is: acf, areas, dens, hex, hist, intervals, neff, pairs, parcoord, recover, rhat, scatter, trace, violin or combo. In particular, combo allows to mix different plot types. For more details see the documentation of the `bayesplot` package, starting from this overview page.
- **pars** An optional character vector of parameter names. If neither pars nor regex_pars is specified, the default is to use all parameters.
- **regex_pars** An optional regular expression to use for parameter selection. Can be specified instead of pars or in addition to pars.
- **include.burnin** A length-one logical vector. If TRUE the burnin iterations (if available) are included in the summary.
- **combo** A character vector providing the plot types to combine (see `mcmc_combo`).
- **...** Further arguments to pass on.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>
plot.dmbc_fit_list,ANY-method

Provide a graphical summary of a dmbc_fit_list class instance.

Description

Provide a graphical summary of a dmbc_fit_list class instance.

Usage

## S4 method for signature 'dmbc_fit_list,ANY'
plot(
  x,
  what = "trace",
  pars = character(),
  regex_pars = "lambda",
  include.burnin = FALSE,
  combo = NULL,
  ...
)

Arguments

x

An object of class dmbc_fit_list.

what

A length-one character vector providing the plot type to produce. Admissible
values are those provided by the bayesplot package, that is: acf, areas, dens,
hex, hist, intervals, neff, pairs, parcoord, recover, rhat, scatter, trace,
violin or combo. In particular, combo allows to mix different plot types. For
more details see the documentation of the bayesplot package, starting from this
overview page.

pars

An optional character vector of parameter names. If neither pars nor regex_pars
is specified, the default is to use all parameters.

regex_pars

An optional regular expression to use for parameter selection. Can be speci-
ified instead of pars or in addition to pars.

include.burnin

A length-one logical vector. If TRUE the burnin iterations (if available) are in-
cluded in the summary.

combo

A character vector providing the plot types to combine (see mcmc_combo).

...

Further arguments to pass on.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>
**plot.dmbc.ic,ANY-method**

*Provide a graphical summary of a dmbc.ic class instance.*

---

**Description**

Provide a graphical summary of a `dmbc.ic` class instance.

**Usage**

```r
## S4 method for signature 'dmbc.ic,ANY'
plot(x, size = NULL, ...)
```

**Arguments**

- `x` An object of class `dmbc.ic`.
- `size` A length-two numeric vector providing the optional sizes of points and lines in the plot.
- `...` Further arguments to pass on (currently ignored).

**Author(s)**

Sergio Venturini <sergio.venturini@unicatt.it>

---

**show,dmbc_config-method**

*Show an instance of the dmbc_config class.*

---

**Description**

Show an instance of the `dmbc_config` class.

**Usage**

```r
## S4 method for signature 'dmbc_config'
show(object)
```

**Arguments**

- `object` An object of class `dmbc_config`.

**Author(s)**

Sergio Venturini <sergio.venturini@unicatt.it>
show.dmbc_data-method

Show an instance of the dmbc_data class.

Description

Show an instance of the dmbc_data class.

Usage

## S4 method for signature 'dmbc_data'
show(object)

Arguments

object An object of class dmbc_data.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

show.dmbc_fit-method

Show an instance of the dmbc_fit class.

Description

Show an instance of the dmbc_fit class.

Usage

## S4 method for signature 'dmbc_fit'
show(object)

Arguments

object An object of class dmbc_fit.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>
show, dmbc_fit_list-method

Show an instance of the dmbc_fit_list class.

Description

Show an instance of the dmbc_fit_list class.

Usage

```r
## S4 method for signature 'dmbc_fit_list'
show(object)
```

Arguments

- object: An object of class `dmbc_fit_list`.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

---

show, dmbc_ic-method

Show an instance of the dmbc_ic class.

Description

Show an instance of the dmbc_ic class.

Usage

```r
## S4 method for signature 'dmbc_ic'
show(object)
```

Arguments

- object: An object of class `dmbc_ic`.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>
show, dmbc_model-method

Show an instance of the dmbc_model class.

Description

Show an instance of the dmbc_model class.

Usage

## S4 method for signature 'dmbc_model'
show(object)

Arguments

object An object of class dmbc_model.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

simdiss

Simulated binary dissimilarity matrices.

Description

A dataset containing a list of simulated binary dissimilarity matrices.

Usage

data(simdiss)

Format

A dmbc_data object whose diss element is a list of 10 binary dissimilarity matrices. Each matrix is defined as a dist object measuring the agreement among 16 different units.

Examples

data(simdiss)
library(bayesplot)
cols <- color_scheme_set("brightblue")
plot(simdiss, colors = unlist(cols)[c(1, 6)], font = 1, cex.font = 0.75)
Description

Subsetting a `dmbc_fit` object.

Usage

```r
## S4 method for signature 'dmbc_fit'
subset(x, pars = character(), regex_pars = character(), ...)
```

Arguments

- `x` An object of class `dmbc_fit`.
- `pars` An optional character vector of parameter names. If neither `pars` nor `regex_pars` is specified, the default is to use all parameters.
- `regex_pars` An optional regular expression to use for parameter selection. Can be specified instead of `pars` or in addition to `pars`.
- `...` Further arguments to pass on (currently ignored).

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

Description

Subsetting a `dmbc_fit_list` object.

Usage

```r
## S4 method for signature 'dmbc_fit_list'
subset(x, pars = character(), regex_pars = character(), ...)
```

Arguments

- `x` An object of class `dmbc_fit_list`.
- `pars` An optional character vector of parameter names. If neither `pars` nor `regex_pars` is specified, the default is to use all parameters.
- `regex_pars` An optional regular expression to use for parameter selection. Can be specified instead of `pars` or in addition to `pars`.
- `...` Further arguments to pass on (currently ignored).
Author(s)
Sergio Venturini <sergio.venturini@unicatt.it>

summary.dmbc_config-method

Provide a summary of a dmbc_config class instance.

Description
Provide a summary of a dmbc_config class instance.

Usage
## S4 method for signature 'dmbc_config'
summary(object)

Arguments
object An object of class dmbc_config.

Author(s)
Sergio Venturini <sergio.venturini@unicatt.it>

summary.dmbc_data-method

Provide a summary of a dmbc_data class instance.

Description
Provide a summary of a dmbc_data class instance.

Usage
## S4 method for signature 'dmbc_data'
summary(object)

Arguments
object An object of class dmbc_data.

Author(s)
Sergio Venturini <sergio.venturini@unicatt.it>
Provide a summary of a \texttt{dmbc\_fit} class instance.

### Usage

```r
## S4 method for signature 'dmbc_fit'
summary(object, include.burnin = FALSE, summary.Z = FALSE, ...)
```

### Arguments

- **object**: An object of class \texttt{dmbc\_fit}.
- **include.burnin**: A length-one logical vector. If TRUE the burnin iterations (if available) are included in the summary.
- **summary.Z**: A length-one logical vector. If TRUE the summary also includes the latent configuration coordinates.
- **...**: Further arguments to pass on (currently ignored).

### Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

Provide a summary of a \texttt{dmbc\_fit\_list} class instance.

### Usage

```r
## S4 method for signature 'dmbc_fit_list'
summary(object, include.burnin = FALSE, summary.Z = FALSE, ...)
```

### Arguments

- **object**: An object of class \texttt{dmbc\_fit\_list}.
- **include.burnin**: A length-one logical vector. If TRUE the burnin iterations (if available) are included in the summary.
- **summary.Z**: A length-one logical vector. If TRUE the summary also includes the latent configuration coordinates.
- **...**: Further arguments to pass on (currently ignored).
**Author(s)**
Sergio Venturini <sergio.venturini@unicatt.it>

---

**summary.dmbo_ic-method**

*Provide a summary of a dmbo_ic class instance.*

---

**Description**
Provide a summary of a dmbo_ic class instance.

**Usage**
```r
## S4 method for signature 'dmbo_ic'
summary(object, p = NULL, G = NULL)
```

**Arguments**
- **object**: An object of class **dmbo_ic**.
- **p**: An optional length-one numeric vector providing the number of latent space dimension to use in the summary.
- **G**: An optional length-one numeric vector providing the number of clusters to use in the summary.

**Author(s)**
Sergio Venturini <sergio.venturini@unicatt.it>

---

**update.dmbo_ic-method**

*Provide an update of a dmbo_ic class instance.*

---

**Description**
Provide an update of a dmbo_ic class instance.

**Usage**
```r
## S4 method for signature 'dmbo_ic'
update(object, pmax = NULL, Gmax = NULL, ...)
```
Arguments

object
  An object of class \code{dmbc_ic}.

pmax
  A length-one numeric vector indicating the maximum number of dimensions of
  the latent space to consider.

Gmax
  A length-one numeric vector indicating the maximum number of cluster to con-
  sider.

... Further arguments to pass on (currently ignored).

Value

A \code{dmbc_ic} object.

Author(s)

Sergio Venturini <sergio.venturini@unicatt.it>

References

Venturini, S., Piccarreta, R. (2021), "A Bayesian Approach for Model-Based Clustering of Several
Binary Dissimilarity Matrices: the \pkg{dmbc} Package in \proglang{R}", Journal of Statistical Software,
100, 16, 1–35, <10.18637/jss.v100.i16>.

See Also

dmbc() for fitting a DMBC model.
dmbc_ic for a description of the elements included in the returned object.

Examples

```r
## Not run:
data(simdiss, package = "dmbc")

pmax <- 2
Gmax <- 2
prm.prop <- list(z = 1.5, alpha = .75)
burnin <- 2000
nsim <- 1000
seed <- 1809
set.seed(seed)

control <- list(burnin = burnin, nsim = nsim, z.prop = prm.prop["z"],
  alpha.prop = prm.prop["alpha"], random.start = TRUE, verbose = TRUE,
  thin = 10, store.burnin = TRUE)
sim.ic <- dmbc_IC(data = simdiss, pmax = pmax, Gmax = Gmax, control = control,
est = "mean")

pmax <- pmax + 1
Gmax <- Gmax + 2
new.ic <- update(sim.ic, pmax = pmax, Gmax = Gmax)
```

new.ic

# plot the results
library(bayesplot)
library(ggplot2)
color_scheme_set("mix-yellow-blue")
p <- plot(new.ic, size = c(4, 1.5))
p + panel_bg(fill = "gray90", color = NA)

## End(Not run)
Index

* based
  dmbc_control, 18
  dmbc_prior, 45

* clustering
  dmbc_control, 18
  dmbc_prior, 45

* datasets
  animals, 5
  kinship, 52
  simdiss, 62

* model
  dmbc_control, 18
  dmbc_prior, 45

adjust_x, 4
animals, 5

bayesplot, 57, 58
bdms, 5, 6, 10, 11, 13, 15, 15, 15
bdms_get_x_mode, 10

check_control (dmbc_control), 18
check_list_na, 11
check_prior (dmbc_prior), 45
clusters, dmbc_config-method, 12
cmdscale, 8
comp_ssr, 13, 54
control_dmbc (dmbc_control), 18

dist, 19, 38
dmbc, 14, 18, 20, 25–27, 36, 39–41, 45, 46, 67
dmbc-internal (dmbc_mcmc), 42
dmbc-p (dmbc-package), 3
dmbc-package, 3
dmbc-pkg (dmbc-package), 3
dmbc_check_groups, 16, 42
dmbc_config, 4, 12, 28, 47, 55, 59, 64
dmbc_config (dmbc_config-class), 17
dmbc_config-class, 17
dmbc_config-clusters
  (clusters, dmbc_config-method), 12

dmbc_config-initialize
  (initialize, dmbc_config-method), 46

dmbc_config-plot
  (plot, dmbc_config, ANY-method), 55

dmbc_config-show
  (show, dmbc_config-method), 59

dmbc_config-summary
  (summary, dmbc_config-method), 64

dmbc_control, 14, 18, 21, 23, 36, 50

dmbc_data, 3, 5, 15, 22, 29, 30, 32, 33, 35, 48
  53, 56, 60, 62, 64

dmbc_data (dmbc_data-class), 20

dmbc_data-class, 20

dmbc_data-initialize
  (initialize, dmbc_data-method), 48

dmbc_data-plot
  (plot, dmbc_data, ANY-method), 56

dmbc_data-show (show, dmbc_data-method), 60

dmbc_data-summary
  (summary, dmbc_data-method), 64

dmbc_fit, 4, 21, 24, 49, 57, 60, 63, 65
dmbc_fit-class, 23

dmbc_fit-initialize
  (initialize, dmbc_fit-method), 48

dmbc_fit-plot
  (plot, dmbc_fit, ANY-method), 57

dmbc_fit-show (show, dmbc_fit-method), 60

dmbc_fit-subset
  (subset, dmbc_fit-method), 63

dmbc_fit-summary
(summary, dmbc_fit-method), 65
dmbc_fit_list, 4, 15, 16, 22, 29, 30, 32, 33, 35, 42, 50, 58, 61, 63, 65
dmbc_fit_list (dmbc_fit_list-class), 24
dmbc_fit_list-class, 24

dmbc_fit_list-initialize
  (initialize, dmbc_fit_list-method), 50
dmbc_fit_list-plot
  (plot, dmbc_fit_list, ANY-method), 58
dmbc_fit_list-show
  (show, dmbc_fit_list-method), 61
dmbc_fit_list-subset
  (subset, dmbc_fit_list-method), 63
dmbc_fit_list-summary
  (summary, dmbc_fit_list-method), 65
dmbc_fit_list_to_list, 25
dmbc_fit_list_to_mcmc.list, 26
dmbc_fit_to_mcmc, 27
dmbc_get_configuration, 16, 28, 42
dmbc_get_map, 29
dmbc_get_ml, 31
dmbc_get_postmean, 32
dmbc_get_postmedian, 34
dmbc IC, 4, 36, 37, 45, 51, 59, 61, 66, 67
dmbc IC (dmbc IC-class), 37
dmbc IC-class, 37
dmbc IC-initialize
  (initialize, dmbc IC-method), 51
dmbc IC-plot
  (plot, dmbc IC, ANY-method), 59
dmbc IC-show
  (show, dmbc IC-method), 61
dmbc IC-summary
  (summary, dmbc IC-method), 66
dmbc IC-update
  (update, dmbc IC-method), 66
dmbc init, 21, 38
dmbc internal (dmbc mcmc), 42
dmbc logLik, 39
dmbc logLik_rbmds, 40
dmbc match groups, 41
dmbc mcmc, 42
dmbc model, 4, 23, 50, 52, 62
dmbc model (dmbc model-class), 44
dmbc model-class, 44
dmbc model-initialize
  (initialize, dmbc model-method), 52
dmbc model-show
  (show, dmbc model-method), 62
dmbc pack par (dmbc mcmc), 42
dmbc prior, 14, 21, 23, 36, 45, 50
dmbc relabel (dmbc mcmc), 42
ggsave, 4
initialize, dmbc config-method, 46
initialize, dmbc data-method, 48
initialize, dmbc fit-method, 48
initialize, dmbc fit list-method, 50
initialize, dmbc IC-method, 51
initialize, dmbc model-method, 52
kinship, 52
mcmc, 27
mcmc.list, 26
mcmc combo, 57, 58
mdsic, 54
parallel, 19
plot, dmbc config, ANY-method, 55
plot, dmbc config-method
  (plot, dmbc config, ANY-method), 55
plot, dmbc data, ANY-method, 56
plot, dmbc data-method
  (plot, dmbc data, ANY-method), 56
plot, dmbc fit, ANY-method, 57
plot, dmbc fit-method
  (plot, dmbc fit, ANY-method), 57
plot, dmbc fit list, ANY-method, 58
plot, dmbc fit list-method
  (plot, dmbc fit list, ANY-method), 58
plot, dmbc IC, ANY-method, 59
plot, dmbc IC-method
  (plot, dmbc IC, ANY-method), 59
prior dmbc (dmbc prior), 45
show, dmbc config-method, 59
show, dmbc data-method, 60
show, dmbc fit-method, 60
show, dmbc fit list-method, 61
INDEX

show, dmbc_ic-method, 61
show, dmbc_model-method, 62
simdiss, 62
subset, dmbc_fit-method, 63
subset, dmbc_fit_list-method, 63
summary, dmbc_config-method, 64
summary, dmbc_data-method, 64
summary, dmbc_fit-method, 65
summary, dmbc_fit_list-method, 65
summary, dmbc_ic-method, 66

theme_default, 4

update, dmbc_ic-method, 66
update_prior (dmbc_prior), 45