Package ‘mclust’

November 20, 2020

Version 5.4.7
Date 2020-11-08
Title Gaussian Mixture Modelling for Model-Based Clustering, Classification, and Density Estimation
Description Gaussian finite mixture models fitted via EM algorithm for model-based clustering, classification, and density estimation, including Bayesian regularization, dimension reduction for visualisation, and resampling-based inference.
Depends R (>= 3.0)
Imports stats, utils, graphics, grDevices
Suggests knitr (>= 1.12), rmarkdown (>= 0.9), mix (>= 1.0), geometry (>= 0.3-6), MASS
License GPL (>= 2)
URL https://mclust-org.github.io/mclust/
VignetteBuilder knitr
Repository CRAN
ByteCompile true
NeedsCompilation yes
LazyData yes
Encoding UTF-8
Author Chris Fraley [aut], Adrian E. Raftery [aut] (<https://orcid.org/0000-0002-6589-301X>), Luca Scrucca [aut, cre] (<https://orcid.org/0000-0003-3826-0484>), Thomas Brendan Murphy [ctb] (<https://orcid.org/0000-0002-5668-7046>), Michael Fop [ctb] (<https://orcid.org/0000-0003-3936-2757>)
Maintainer Luca Scrucca <luca.scrucca@unipg.it>
Date/Publication 2020-11-20 10:40:03 UTC
R topics documented:

mclust-package ........................................... 4
acidity .................................................... 5
adjustedRandIndex ........................................ 6
banknote .................................................... 7
Baudry_etal_2010_JCGS_examples ......................... 8
bic ............................................................ 9
BrierScore ................................................ 10
cdens ....................................................... 12
cdensE ..................................................... 14
cdfMclust .................................................. 15
chevron ..................................................... 17
classError ................................................ 17
classPriorProbs .......................................... 18
clPairs .................................................... 21
clustCombi ................................................ 22
clustCombiOptim ......................................... 25
combiPlot ................................................ 26
combiTree ............................................... 28
combMat .................................................. 29
coordProj ................................................ 30
covw ....................................................... 32
cross ........................................................ 33
cvMclustDA ............................................... 33
decomp2sigma ............................................ 35
defaultPrior ............................................. 36
dens ......................................................... 38
densityMclust ............................................ 39
densityMclust.diagnostic ................................ 40
diabetes ................................................... 42
dmvnorm ................................................... 43
dmnorm .................................................... 44
dmnorm .................................................... 44
dmnorm .................................................... 44
dmnorm .................................................... 44
em ........................................................ 46
dens ......................................................... 38
densityMclust ............................................ 39
densityMclust.diagnostic ................................ 40
diabetes ................................................... 42
dmvnorm ................................................... 43
dmnorm .................................................... 44
dmnorm .................................................... 44
dmnorm .................................................... 44
em ........................................................ 46
dens ......................................................... 38
densityMclust ............................................ 39
densityMclust.diagnostic ................................ 40
diabetes ................................................... 42
dmvnorm ................................................... 43
dmnorm .................................................... 44
dmnorm .................................................... 44
dmnorm .................................................... 44
em ........................................................ 46
dens ......................................................... 38
densityMclust ............................................ 39
densityMclust.diagnostic ................................ 40
diabetes ................................................... 42
dmvnorm ................................................... 43
dmnorm .................................................... 44
dmnorm .................................................... 44
dmnorm .................................................... 44
em ........................................................ 46
dens ......................................................... 38
densityMclust ............................................ 39
densityMclust.diagnostic ................................ 40
diabetes ................................................... 42
dmvnorm ................................................... 43
dmnorm .................................................... 44
dmnorm .................................................... 44
dmnorm .................................................... 44
em ........................................................ 46
dens ......................................................... 38
densityMclust ............................................ 39
densityMclust.diagnostic ................................ 40
diabetes ................................................... 42
dmvnorm ................................................... 43
dmnorm .................................................... 44
dmnorm .................................................... 44
dmnorm .................................................... 44
em ........................................................ 46
dens ......................................................... 38
densityMclust ............................................ 39
densityMclust.diagnostic ................................ 40
diabetes ................................................... 42
mclust-package

Gaussian Mixture Modelling for Model-Based Clustering, Classification, and Density Estimation

Description

Finite Gaussian mixture modelling fitted via EM algorithm for model-based clustering, classification, and density estimation, including Bayesian regularization and dimension reduction.

Details

For a quick introduction to mclust see the vignette A quick tour of mclust.

Author(s)

Chris Fraley, Adrian Raftery and Luca Scrucca.

Maintainer: Luca Scrucca <luca.scrucca@unipg.it>
References

Scrucca L., Fop M., Murphy T. B. and Raftery A. E. (2016) mclust 5: clustering, classification and
density estimation using Gaussian finite mixture models, The R Journal, 8/1, pp. 289-317.

Fraley C. and Raftery A. E. (2002) Model-based clustering, discriminant analysis and density esti-

Fraley C., Raftery A. E., Murphy T. B. and Scrucca L. (2012) mclust Version 4 for R: Normal
Mixture Modeling for Model-Based Clustering, Classification, and Density Estimation. Technical
Report No. 597, Department of Statistics, University of Washington.

Examples

# Clustering
mod1 <- Mclust(iris[,1:4])
summary(mod1)
plot(mod1, what = c("BIC", "classification"))

# Classification
data(banknote)
mod2 <- MclustDA(banknote[,2:7], banknote$Status)
summary(mod2)
plot(mod2)

# Density estimation
mod3 <- densityMclust(faithful$waiting)
summary(mod3)
plot(mod3, faithful$waiting)

---

acidity

Acidity data

Description

Acidity index measured in a sample of 155 lakes in the Northeastern United States. The data are on
the log scale, as analysed by Crawford et al. (1992, 1994). The data were also used to fit mixture
of gaussian distributions by Richardson and Green (1997), and by McLachlan and Peel (2000, Sec.
6.6.2).

Usage

data(acidity)

Source

http://www.stats.bris.ac.uk/~peter/mixdata
adjustedRandIndex

References


adjustedRandIndex  Adjusted Rand Index

Description

Computes the adjusted Rand index comparing two classifications.

Usage

adjustedRandIndex(x, y)

Arguments

x  A numeric or character vector of class labels.

y  A numeric or character vector of class labels. The length of y should be the same as that of x.

Value

The adjusted Rand index comparing the two partitions (a scalar). This index has zero expected value in the case of random partition, and it is bounded above by 1 in the case of perfect agreement between two partitions.

References


See Also

classError, mapClass, table
Examples

```r
a <- rep(1:3, 3)
a
b <- rep(c("A", "B", "C"), 3)
b
adjustedRandIndex(a, b)

a <- sample(1:3, 9, replace = TRUE)
a
b <- sample(c("A", "B", "C"), 9, replace = TRUE)
b
adjustedRandIndex(a, b)
```

```r
irisHCvvv <- hc(modelName = "VVV", data = iris[, -5])
c13 <- hclass(irisHCvvv, 3)
adjustedRandIndex(c13, iris[, 5])
```

```r
irisBIC <- mclustBIC(iris[, -5])
adjustedRandIndex(summary(irisBIC, iris[, -5])$classification, iris[, 5])
adjustedRandIndex(summary(irisBIC, iris[, -5], G = 3)$classification, iris[, 5])
```

---

**banknote**

*Swiss banknotes data*

### Description

The data set contains six measurements made on 100 genuine and 100 counterfeit old-Swiss 1000-franc bank notes.

### Usage

```r
data(banknote)
```

### Format

A data frame with the following variables:

- **Status** the status of the banknote: genuine or counterfeit
- **Length** Length of bill (mm)
- **Left** Width of left edge (mm)
- **Right** Width of right edge (mm)
- **Bottom** Bottom margin width (mm)
- **Top** Top margin width (mm)
- **Diagonal** Length of diagonal (mm)
Simulated Example Datasets From Baudry et al. (2010)

Description

Simulated datasets used in Baudry et al. (2010) to illustrate the proposed mixture components combining method for clustering.

Please see the cited article for a detailed presentation of these datasets. The data frame with name exN.M is presented in Section N.M in the paper.

Test1D (not in the article) has been simulated from a Gaussian mixture distribution in R.

ex4.1 and ex4.2 have been simulated from a Gaussian mixture distribution in R^2.

ex4.3 has been simulated from a mixture of a uniform distribution on a square and a spherical Gaussian distribution in R^2.

ex4.4.1 has been simulated from a Gaussian mixture model in R^2

ex4.4.2 has been simulated from a mixture of two uniform distributions in R^3.

Usage

data(Baudry_etal_2010_JCGS_examples)

Format

ex4.1 is a data frame with 600 observations on 2 real variables.

ex4.2 is a data frame with 600 observations on 2 real variables.

ex4.3 is a data frame with 200 observations on 2 real variables.

ex4.4.1 is a data frame with 800 observations on 2 real variables.

ex4.4.2 is a data frame with 300 observations on 3 real variables.

Test1D is a data frame with 200 observations on 1 real variable.

References

Examples

## Not run:
data(Baudry_etal_2010_JCGS_examples)

output <- clustCombi(data = ex4.4.1)
output # is of class clustCombi

# plots the hierarchy of combined solutions, then some "entropy plots" which
# may help one to select the number of classes
plot(output)

## End(Not run)

bic | BIC for Parameterized Gaussian Mixture Models

Description

Computes the BIC (Bayesian Information Criterion) for parameterized mixture models given the
loglikelihood, the dimension of the data, and number of mixture components in the model.

Usage

bic(modelName, loglik, n, d, G, noise=FALSE, equalPro=FALSE, ...)

Arguments

modelName A character string indicating the model. The help file for mclustModelNames
describes the available models.

loglik The log-likelihood for a data set with respect to the Gaussian mixture model
specified in the modelName argument.

n The number of observations in the data used to compute loglik.

d The dimension of the data used to compute loglik.

G The number of components in the Gaussian mixture model used to compute
loglik.

noise A logical variable indicating whether or not the model includes an optional Poisson
noise component. The default is to assume no noise component.

equalPro A logical variable indicating whether or not the components in the model are
assumed to be present in equal proportion. The default is to assume unequal
mixing proportions.

... Catches unused arguments in an indirect or list call via do.call.

Value

The BIC or Bayesian Information Criterion for the given input arguments.
Brier Score

Brier score to assess the accuracy of probabilistic predictions

Description

The Brier score is a proper score function that measures the accuracy of probabilistic predictions.

Usage

BrierScore(z, class)

Arguments

z

a matrix containing the predicted probabilities of each observation to be classified in one of the classes. Thus, the number of rows must match the length of class, and the number of columns the number of known classes.

class

a numeric, character vector or factor containing the known class labels for each observation. If class is a factor, the number of classes is nlevels(class) with classes levels(class). If class is a numeric or character vector, the number of classes is equal to the number of classes obtained via unique(class).

Details

The Brier Score is the mean square difference between the true classes and the predicted probabilities.

This function implements the original multi-class definition by Brier (1950), normalized to $[0, 1]$ as in Kruppa et al (2014). The formula is the following:

$$BS = \frac{1}{2n} \sum_{i=1}^{n} \sum_{k=1}^{K} (C_{ik} - p_{ik})^2$$

See Also

mclustBIC, nVarParams, mclustModelNames.

Examples

## Not run:

n <- nrow(iris)
d <- ncol(iris)-1
G <- 3

ingEst <- me(modelName="VVI", data=iris[, -5], unmap(iris[, 5]))
names(emEst)

args(bic)
bic(modelName="VVI", loglik=emEst$loglik, n=n, d=d, G=G)
# do.call("bic", emEst)  ## alternative call

## End(Not run)
where \( n \) is the number of observations, \( K \) the number of classes, \( C_{ik} = \{0, 1\} \) the indicator of class \( k \) for observation \( i \), and \( p_{ik} \) is the predicted probability of observation \( i \) to belong to class \( k \).

The above formulation is applicable to multi-class predictions, including the binary case. A small value of the Brier Score indicates high prediction accuracy.

The Brier Score is a strictly proper score (Gneiting and Raftery, 2007), which means that it takes its minimal value only when the predicted probabilities match the empirical probabilities.

References


See Also

cvMclustDA

Examples

```r
# multi-class case
class <- factor(c(5,5,5,2,5,3,1,2,1,1), levels = 1:5)
probs <- matrix(c(0.15, 0.01, 0.08, 0.23, 0.01, 0.23, 0.59, 0.02, 0.38, 0.45,
                  0.36, 0.05, 0.30, 0.46, 0.15, 0.13, 0.06, 0.19, 0.27, 0.17,
                  0.40, 0.34, 0.18, 0.04, 0.47, 0.34, 0.32, 0.01, 0.03, 0.11,
                  0.04, 0.04, 0.09, 0.05, 0.28, 0.27, 0.02, 0.03, 0.12, 0.25,
                  0.05, 0.56, 0.35, 0.22, 0.09, 0.03, 0.01, 0.75, 0.20, 0.02),
                  nrow = 10, ncol = 5)
cbind(class, probs, map = map(probs))
BrierScore(probs, class)

# two-class case
class <- factor(c(1,1,1,2,2,1,1,2,1,1), levels = 1:2)
probs <- matrix(c(0.91, 0.4, 0.56, 0.27, 0.37, 0.7, 0.97, 0.22, 0.68, 0.43,
                  0.09, 0.6, 0.44, 0.73, 0.63, 0.3, 0.03, 0.78, 0.32, 0.57),
                  nrow = 10, ncol = 2)
cbind(class, probs, map = map(probs))
BrierScore(probs, class)

# two-class case when predicted probabilities are constrained to be equal to # 0 or 1, then the (normalized) Brier Score is equal to the classification # error rate
probs <- ifelse(probs > 0.5, 1, 0)
cbind(class, probs, map = map(probs))
BrierScore(probs, class)
classError(map(probs), class)$errorRate
```
# plot Brier score for predicted probabilities in range \([0,1]\)
class <- factor(rep(1, each = 100), levels = 0:1)
prob <- seq(0, 1, by = 0.01)
brier <- sapply(prob, function(p)
    ( z <- matrix(c(1-p,p), nrow = length(class), ncol = 2, byrow = TRUE)
    BrierScore(z, class))
)
plot(prob, brier, type = "l", main = "Scoring all one class",
    xlab = "Predicted probability", ylab = "Brier score")

# brier score for predicting balanced data with constant prob
class <- factor(rep(c(1,0), each = 50), levels = 0:1)
prob <- seq(0, 1, by = 0.01)
brier <- sapply(prob, function(p)
    ( z <- matrix(c(1-p,p), nrow = length(class), ncol = 2, byrow = TRUE)
    BrierScore(z, class))
)
plot(prob, brier, type = "l", main = "Scoring balanced classes",
    xlab = "Predicted probability", ylab = "Brier score")

# brier score for predicting unbalanced data with constant prob
class <- factor(rep(c(0,1), times = c(90,10)), levels = 0:1)
prob <- seq(0, 1, by = 0.01)
brier <- sapply(prob, function(p)
    ( z <- matrix(c(1-p,p), nrow = length(class), ncol = 2, byrow = TRUE)
    BrierScore(z, class))
)
plot(prob, brier, type = "l", main = "Scoring unbalanced classes",
    xlab = "Predicted probability", ylab = "Brier score")

---

cdens  

**Component Density for Parameterized MVN Mixture Models**

**Description**

Computes component densities for observations in MVN mixture models parameterized by eigenvalue decomposition.

**Usage**

cdens(modelName, data, logarithm = FALSE, parameters, warn = NULL, ...)

**Arguments**

- **modelName**  A character string indicating the model. The help file for `mclustModelNames` describes the available models.
- **data**  A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
cdens

logarithm  A logical value indicating whether or not the logarithm of the component densities should be returned. The default is to return the component densities, obtained from the log component densities by exponentiation.

parameters  The parameters of the model:

mean  The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.

variance  A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.

warn  A logical value indicating whether or not a warning should be issued when computations fail. The default is warn=FALSE.

...  Catches unused arguments in indirect or list calls via do.call.

Value  A numeric matrix whose [i,k]th entry is the density or log density of observation i in component k. The densities are not scaled by mixing proportions.

Note  When one or more component densities are very large in magnitude, it may be possible to compute the logarithm of the component densities but not the component densities themselves due to overflow.

See Also  cdensE, ..., cdensVVV, dens, estep, mclustModelNames, mclustVariance, mclust.options, do.call

Examples  

z2 <- unmap(hclass(hcVVV(faithful),2)) # initial value for 2 class case

model <- me(modelName = "EEE", data = faithful, z = z2)

cdens(modelName = "EEE", data = faithful, logarithm = TRUE,
  parameters = model$parameters)[1:5,]

data(cross)

odd <- seq(1, nrow(cross), by = 2)
oddBIC <- mclustBIC(cross[odd,-1])
oddModel <- mclustModel(cross[odd,-1], oddBIC) ## best parameter estimates

names(oddModel)

ev <- odd + 1

densities <- cdens(modelName = oddModel$modelName, data = cross[even,-1],
  parameters = oddModel$parameters)

cbind(class = cross[even,1], densities)[1:5,]
Component Density for a Parameterized MVN Mixture Model

Description

Computes component densities for points in a parameterized MVN mixture model.

Usage

\[
\text{cdensE}(\text{data}, \text{logarithm} = \text{FALSE}, \text{parameters}, \text{warn} = \text{NULL}, \ldots)
\]

Arguments

- **data**: A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- **logarithm**: A logical value indicating whether or not the logarithm of the component densities should be returned. The default is to return the component densities, obtained from the log component densities by exponentiation.
- **parameters**: The parameters of the model:
  - **mean**: The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.
  - **variance**: A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
Mixing proportions for the components of the mixture. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.

A logical value indicating whether or not a warning should be issued when computations fail. The default is warn=FALSE.

Catches unused arguments in indirect or list calls via do.call.

A numeric matrix whose \([i,j]\)th entry is the density of observation \(i\) in component \(j\). The densities are not scaled by mixing proportions.

When one or more component densities are very large in magnitude, then it may be possible to compute the logarithm of the component densities but not the component densities themselves due to overflow.

See Also
cdens, dens, mclustVariance, mstep, mclust.options, do.call.

Examples

```r
## Not run:
z2 <- unmap(hclass(hcVVV(faithful),2)) # initial value for 2 class case
model <- meVVV(data=faithful, z=z2)
cdensVVV(data=faithful, logarithm = TRUE, parameters = model$parameters)
data(cross)
z2 <- unmap(cross[,1])
model <- meEEV(data = cross[,1], z = z2)
EEVdensities <- cdensEEV( data = cross[,1], parameters = model$parameters)
cbind(cross[,1],map(EEVdensities))
## End(Not run)
```

Compute the cumulative density function (cdf) or quantiles from an estimated one-dimensional Gaussian mixture fitted using densityMclust.
Usage

cdfMclust(object, data, ngrid = 100, ...)
quantileMclust(object, p, ...)

Arguments

object  a `densityMclust` model object.
data  a numeric vector of evaluation points.
ngrid  the number of points in a regular grid to be used as evaluation points if no data
       are provided.
p  a numeric vector of probabilities.
...  further arguments passed to or from other methods.

Details

The cdf is evaluated at points given by the optional argument `data`. If not provided, a regular grid
of length `ngrid` for the evaluation points is used.
The quantiles are computed using interpolating splines on an adaptive finer grid.

Value

cdfMclust returns a list of x and y values providing, respectively, the evaluation points and the
estimated cdf.
quantileMclust returns a vector of quantiles.

Author(s)

Luca Scrucca

See Also
densityMclust, plot.densityMclust.

Examples

```r
x <- c(rnorm(100), rnorm(100, 3, 2))
dens <- densityMclust(x)
summary(dens, parameters = TRUE)
cdf <- cdfMclust(dens)
str(cdf)
q <- quantileMclust(dens, p = c(0.01, 0.1, 0.5, 0.9, 0.99))
cbind(quantile = q, cdf = cdfMclust(dens, q)$y)
plot(cdf, type = "l", xlab = "x", ylab = "CDF")
points(q, cdfMclust(dens, q)$y, pch = 20, col = "red3")

par(mfrow = c(2,2))
dens.waiting <- densityMclust(faithful$waiting)
plot(dens.waiting)
```
plot(cdfMclust(dens.waiting), type = "l", 
    xlab = dens.waiting$varname, ylab = "CDF")
dens.eruptions <- densityMclust(faithful$eruptions)
plot(dens.eruptions)
plot(cdfMclust(dens.eruptions), type = "l", 
    xlab = dens.eruptions$varname, ylab = "CDF")
par(mfrow = c(1,1))

---

**chevron**

*Simulated minefield data*

**Description**

A set of simulated bivariate minefield data (1104 observations).

**Usage**

data(chevron)

**References**


---

**classError**

*Classification error*

**Description**

Computes the error rate of a given classification relative to the known classes, and the location of 
misclassified data points.

**Usage**

classError(classification, class)

**Arguments**

- **classification**: A numeric, character vector or factor specifying the predicted class labels. Must 
have the same length as class.
- **class**: A numeric, character vector or factor of known true class labels. Must have the 
same length as classification.
classPriorProbs

Details

If more than one mapping between predicted classification and the known truth corresponds to the minimum number of classification errors, only one possible set of misclassified observations is returned.

Value

A list with the following two components:

- `misclassified`: The indexes of the misclassified data points in a minimum error mapping between the predicted classification and the known true classes.
- `errorRate`: The error rate corresponding to a minimum error mapping between the predicted classification and the known true classes.

See Also

`map`, `mapClass`, `table`

Examples

```r
(a <- rep(1:3, 3))
(b <- rep(c("A", "B", "C"), 3))
classError(a, b)

(a <- sample(1:3, 9, replace = TRUE))
(b <- sample(c("A", "B", "C"), 9, replace = TRUE))
classError(a, b)

class <- factor(c(5,5,5,2,5,3,1,2,1,1), levels = 1:5)
probs <- matrix(c(0.15, 0.01, 0.08, 0.23, 0.01, 0.23, 0.59, 0.02, 0.38, 0.45,
                  0.36, 0.05, 0.30, 0.46, 0.15, 0.13, 0.06, 0.19, 0.27, 0.17,
                  0.40, 0.34, 0.18, 0.04, 0.47, 0.34, 0.32, 0.01, 0.03, 0.11,
                  0.04, 0.04, 0.09, 0.05, 0.28, 0.27, 0.02, 0.03, 0.12, 0.25,
                  0.05, 0.56, 0.35, 0.22, 0.09, 0.03, 0.01, 0.75, 0.20, 0.02),
                  nrow = 10, ncol = 5)
cbind(class, probs, map = map(probs))
classError(map(probs), class)
```

classPriorProbs

Estimation of class prior probabilities by EM algorithm

Description

A simple procedure to improve the estimation of class prior probabilities when the training data does not reflect the true a priori probabilities of the target classes. The EM algorithm used is described in Saerens et al (2002).
Usage

classPriorProbs(object, newdata = object$data,
        itmax = 1e3, eps = sqrt(.Machine$double.eps))

Arguments

object an object of class 'MclustDA' resulting from a call to MclustDA.
newdata a data frame or matrix giving the data. If missing the train data obtained from
the call to MclustDA are used.
itmax an integer value specifying the maximal number of EM iterations.
eps a scalar specifying the tolerance associated with deciding when to terminate the
EM iterations.

Details

The estimation procedure employes an EM algorithm as described in Saerens et al (2002).

Value

A vector of class prior estimates which can then be used in the predict.MclustDA to improve
predictions.

References

priori probabilities: a simple procedure, Neural computation, 14 (1), 21–41.

See Also

MclustDA, predict.MclustDA

Examples

## Not run:
# generate data from a mixture f(x) = 0.9 * N(0,1) + 0.1 * N(3,1)
n <- 10000
mixpro <- c(0.9, 0.1)
class <- factor(sample(0:1, size = n, prob = mixpro, replace = TRUE))
x <- ifelse(class == 1, rnorm(n, mean = 3, sd = 1),
             rnorm(n, mean = 0, sd = 1))

density(x[class==0], breaks = 11, xlab = "x",
        col = adjustcolor("dodgerblue2", alpha.f = 0.5), border = "white")

density(x[class==1], breaks = 11, add = TRUE,
        col = adjustcolor("red3", alpha.f = 0.5), border = "white")

# generate training data from a balanced case-control sample, i.e.
# f(x) = 0.5 * N(0,1) + 0.5 * N(3,1)
n_train <- 1000
class_train <- factor(sample(0:1, size = n_train, prob = c(0.5, 0.5), replace = TRUE))
x_train <- ifelse(class_train == 1, rnorm(n_train, mean = 3, sd = 1),
                 rnorm(n_train, mean = 0, sd = 1))

hist(x_train[class_train==0], breaks = 11, xlim = range(x_train),
     main = "", xlab = "x",
     col = adjustcolor("dodgerblue2", alpha.f = 0.5), border = "white")
hist(x_train[class_train==1], breaks = 11, add = TRUE,
     col = adjustcolor("red3", alpha.f = 0.5), border = "white")
box()

# fit a MclustDA model
mod <- MclustDA(x_train, class_train)
summary(mod, parameters = TRUE)

# test set performance
pred <- predict(mod, newdata = x)
classError(pred$classification, class)$error
BrierScore(pred$z, class)

# compute performance over a grid of prior probs
priorProp <- seq(0.01, 0.99, by = 0.01)
CE <- BS <- rep(as.double(NA), length(priorProp))
for(i in seq(priorProp))
{
  pred <- predict(mod, newdata = x, prop = c(1-priorProp[i], priorProp[i]))
  CE[i] <- classError(pred$classification, class = class)$error
  BS[i] <- BrierScore(pred$z, class)
}

# estimate the optimal class prior probs
(priorProbs <- classPriorProbs(mod, x))
pred <- predict(mod, newdata = x, prop = priorProbs)

# Summary of results:
priorProp[which.min(CE)] # best prior of class 1 according to classification error
priorProp[which.min(BS)] # best prior of class 1 according to Brier score
priorProbs # optimal estimated class prior probabilities
**clPairs**

**Pairwise Scatter Plots showing Classification**

## Description

Creates a scatter plot for each pair of variables in given data. Observations in different classes are represented by different colors and symbols.

## Usage

```r
clPairs(data, classification,
symbols = NULL, colors = NULL, cex = NULL,
labels = dimnames(data)[[2]], cex.labels = 1.5,
gap = 0.2, grid = FALSE, ...)

clPairsLegend(x, y, class, col, pch, cex, box = TRUE, ...)
```

## Arguments

- `data`: A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- `classification`: A numeric or character vector representing a classification of observations (rows) of data.
- `symbols`: Either an integer or character vector assigning a plotting symbol to each unique class in `classification`. Elements in `symbols` correspond to classes in order of appearance in the sequence of observations (the order used by the function `unique`). The default is given by `mclust.options("classPlotSymbols")`.
- `colors`: Either an integer or character vector assigning a color to each unique class in `classification`. Elements in `colors` correspond to classes in order of appearance in the sequence of observations (the order used by the function `unique`). The default is given by `mclust.options("classPlotColors")`.
- `cex`: A vector of numerical values specifying the size of the plotting symbol for each unique class in `classification`. Values in `cex` correspond to classes in order of appearance in the sequence of observations (the order used by the function `unique`). By default `cex = 1` for all classes is used.
- `labels`: A vector of character strings for labelling the variables. The default is to use the column dimension names of `data`.
- `cex.labels`: A numerical value specifying the size of the text labels.
- `gap`: An argument specifying the distance between subplots (see `pairs`).
- `grid`: A logical specifying if grid lines should be added to panels (see `grid`).
x, y

The x and y co-ordinates with respect to a graphic device having plotting region coordinates `par("usr" = c(0,1,0,1)).

class

The class labels.

box

A logical, if TRUE then a box is drawn around the current plot figure.

col, pch

The colors and plotting symbols appearing in the legend.

... For a `clPairs` call may be additional arguments to be passed to `pairs`. For a `clPairsLegend` call may be additional arguments to be passed to `legend`.

Details

The function `clPairs()` draws scatter plots on the current graphics device for each combination of variables in data. Observations of different classifications are labeled with different symbols.

The function `clPairsLegend()` can be used to add a legend. See examples below.

Value

The function `clPairs()` invisibly returns a list with the following components:

class A character vector of class labels.

col A vector of colors used for each class.

pch A vector of plotting symbols used for each class.

See Also

`pairs`, `coordProj`, `mclust.options`

Examples

clPairs(iris[,1:4], cl = iris$Species)

clp <- clPairs(iris[,1:4], cl = iris$Species, lower.panel = NULL)
clPairsLegend(0.1, 0.4, class = clp$class,
            col = clp$col, pch = clp$pch,
            title = "Iris data")

---

clustCombi

Combining Gaussian Mixture Components for Clustering

Description

Provides a hierarchy of combined clusterings from the EM/BIC Gaussian mixture solution to one class, following the methodology proposed in the article cited in the references.

Usage

clustCombi(object = NULL, data = NULL, ...)
Arguments

object: An object returned by Mclust giving the optimal (according to BIC) parameters, conditional probabilities, and log-likelihood, together with the associated classification and its uncertainty. If not provided, the data argument must be specified.

data: A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables. If the object argument is not provided, the function Mclust is applied to the given data to fit a mixture model.

...: Optional arguments to be passed to called functions. Notably, any argument (such as the numbers of components for which the BIC is computed; the models to be fitted by EM; initialization parameters for the EM algorithm, etc.) to be passed to Mclust in case object = NULL. Please see the Mclust documentation for more details.

Details

Mclust provides a Gaussian mixture fitted to the data by maximum likelihood through the EM algorithm, for the model and number of components selected according to BIC. The corresponding components are hierarchically combined according to an entropy criterion, following the methodology described in the article cited in the references section. The solutions with numbers of classes between the one selected by BIC and one are returned as a clustCombi class object.

Value

A list of class clustCombi giving the hierarchy of combined solutions from the number of components selected by BIC to one. The details of the output components are as follows:

classification: A list of the data classifications obtained for each combined solution of the hierarchy through a MAP assignment.

combiM: A list of matrices. combiM[[K]] is the matrix used to combine the components of the (K+1)-classes solution to get the K-classes solution. Please see the examples.

combiz: A list of matrices. combiz[[K]] is a matrix whose [i,k]th entry is the probability that observation i in the data belongs to the kth class according to the K-classes combined solution.

MclustOutput: A list of class Mclust. Output of a call to the Mclust function (as provided by the user or the result of a call to the Mclust function) used to initiate the combined solutions hierarchy: please see the Mclust function documentation for details.

Author(s)

J.-P. Baudry, A. E. Raftery, L. Scrucca

References

See Also

plot.clustCombi

Examples

data(Baudry_etal_2010_JCGS_examples)

# run Mclust using provided data
output <- clustCombi(data = ex4.1)
## Not run:
# or run Mclust and then clustcombi on the returned object
mod <- Mclust(ex4.1)
output <- clustCombi(mod)
## End(Not run)

output
summary(output)

## Not run:
# run Mclust using provided data and any further optional argument provided
output <- clustCombi(data = ex4.1, modelName = "EEV", G = 1:15)
## End(Not run)

# plot the hierarchy of combined solutions
plot(output, what = "classification")
# plot some "entropy plots" which may help one to select the number of classes
plot(output, what = "entropy")
# plot the tree structure obtained from combining mixture components
plot(output, what = "tree")

# the selected model and number of components obtained from Mclust using BIC
output$MclustOutput

# the matrix whose [i,k]th entry is the probability that i-th observation in
# the data belongs to the k-th class according to the BIC solution
head(output$combiz[[output$MclustOutput$G]])

# the matrix whose [i,k]th entry is the probability that i-th observation in
# the data belongs to the k-th class according to the first combined solution
head(output$combiz[[output$MclustOutput$G-1]])

# the matrix describing how to merge the 6-classes solution to get the
# 5-classes solution
output$combiM[[5]]

# for example the following code returns the label of the class (in the
# 5-classes combined solution) to which the 4th class (in the 6-classes
# solution) is assigned. Only two classes in the (K+1)-classes solution
# are assigned the same class in the K-classes solution: the two which
# are merged at this step
output$combiM[[5]]

# recover the 5-classes soft clustering from the 6-classes soft clustering
# and the 6 -> 5 combining matrix
# the hard clustering under the 5-classes solution
head( output$classification[[5]] )

---

**clustCombiOptim**

*Optimal number of clusters obtained by combining mixture components*

### Description

Return the optimal number of clusters by combining mixture components based on the entropy method discussed in the reference given below.

### Usage

```r
clustCombiOptim(object, reg = 2, plot = FALSE, ...)
```

### Arguments

- **object**
  An object of class 'clustCombi' resulting from a call to `clustCombi`.

- **reg**
  The number of parts of the piecewise linear regression for the entropy plots. Choose 2 for a two-segment piecewise linear regression model (i.e. 1 change-point), and 3 for a three-segment piecewise linear regression model (i.e. 3 change-points).

- **plot**
  Logical, if TRUE an entropy plot is also produced.

- **...**
  Further arguments passed to or from other methods.

### Value

The function returns a list with the following components:

- **numClusters.combi**
  The estimated number of clusters.

- **z.combi**
  A matrix whose $i,k$th entry is the probability that observation $i$ in the data belongs to the $k$th cluster.

- **cluster.combi**
  The clustering labels.

### Author(s)

J.-P. Baudry, A. E. Raftery, L. Scrucca

### References

combiPlot

Plot Classifications Corresponding to Successive Combined Solutions

Description

Plot classifications corresponding to successive combined solutions.

Usage

combiPlot(data, z, combiM, ...)

Arguments

data The data.

z A matrix whose [i,k]th entry is the probability that observation i in the data belongs to the kth class, for the initial solution (ie before any combining). Typically, the one returned by Mclust/BIC.

combiM A "combining matrix" (as provided by clustCombi), i.e. a matrix whose kth row contains only zeros, but in columns corresponding to the labels of the classes in the initial solution to be merged together to get the combined solution.

... Other arguments to be passed to the Mclust plot functions.

Value

Plot the classifications obtained by MAP from the matrix t(combiM %*% t(z)), which is the matrix whose [i,k]th entry is the probability that observation i in the data belongs to the kth class, according to the combined solution obtained by merging (according to combiM) the initial solution described by z.
combiPlot

Author(s)

J.-P. Baudry, A. E. Raftery, L. Scrucca

References


See Also

clustCombi, combMat, clustCombi

Examples

```r
## Not run:
data(Baudry_etal_2010_JCGS_examples)
MclustOutput <- Mclust(ex4.1)
MclustOutput$G # Mclust/BIC selected 6 classes

par(mfrow=c(2,2))

combiM0 <- diag(6) # is the identity matrix
# no merging: plot the initial solution, given by z
combiPlot(ex4.1, MclustOutput$z, combiM0, cex = 3)
title("No combining")

combiM1 <- combMat(6, 1, 2) # let's merge classes labeled 1 and 2
combiM1
combiPlot(ex4.1, MclustOutput$z, combiM1)
title("Combine 1 and 2")

# let's merge classes labeled 1 and 2, and then components labeled (in this
# new 5-classes combined solution) 1 and 2
combiM2 <- combMat(5, 1, 2) %*% combMat(6, 1, 2)
combiM2
combiPlot(ex4.1, MclustOutput$z, combiM2)
title("Combine 1, 2 and then 1 and 2 again")

plot(0,0,type="n", xlab = "", ylab = "", axes = FALSE)
legend("center", legend = 1:6,
      col = mclust.options("classPlotColors"),
      pch = mclust.options("classPlotSymbols"),
      title = "Class labels:")
## End(Not run)
```
combiTree

Tree structure obtained from combining mixture components

Description
The method implemented in `clustCombi` can be used for combining Gaussian mixture components for clustering. This provides a hierarchical structure which can be graphically represented as a tree.

Usage

```r
combiTree(object, type = c("triangle", "rectangle"),
          yaxis = c("entropy", "step"),
          edgePar = list(col = "darkgray", lwd = 2),
          ...)
```

Arguments

- **object**: An object of class '\code{clustCombi}' resulting from a call to `clustCombi`.
- **type**: A string specifying the dendrogram's type. Possible values are "triangle" (default), and "rectangle".
- **yaxis**: A string specifying the quantity used to draw the vertical axis. Possible values are "entropy" (default), and "step".
- **edgePar**: A list of plotting parameters. See `dendrogram`.
- **...**: Further arguments passed to or from other methods.

Value
The function always draw a tree and invisibly returns an object of class '\code{dendrogram}' for fine tuning.

Author(s)
L. Scrucca

See Also
`clustCombi`

Examples
```
## Not run:
data(Baudry_etal_2010_JCGS_examples)
output <- clustCombi(data = ex4.1)
combiTree(output)
combiTree(output, type = "rectangle")
combiTree(output, yaxis = "step")
combiTree(output, type = "rectangle", yaxis = "step")
```
## combMat

### Combining Matrix

**Description**

Create a combining matrix

**Usage**

`combMat(K, l1, l2)`

**Arguments**

- **K**: The original number of classes: the matrix will define a combining from K to (K-1) classes.
- **l1**: Label of one of the two classes to be combined.
- **l2**: Label of the other class to be combined.

**Value**

If `z` is a vector (length `K`) whose `k`th entry is the probability that an observation belongs to the `k`th class in a `K`-classes classification, then `combiM %*% z` is the vector (length `K-1`) whose `k`th entry is the probability that the observation belongs to the `k`th class in the `K-1`-classes classification obtained by merging classes `l1` and `l2` in the initial classification.

**Author(s)**

J.-P. Baudry, A. E. Raftery, L. Scrucca

**See Also**

`clustCombi`, `combiPlot`
**coordProj**

**Coordinate projections of multidimensional data modeled by an MVN mixture.**

**Description**

Plots coordinate projections given multidimensional data and parameters of an MVN mixture model for the data.

**Usage**

```r
coordProj(data, dimens = c(1,2), parameters = NULL, z = NULL, classification = NULL, truth = NULL, uncertainty = NULL, what = c("classification", "error", "uncertainty"), addEllipses = TRUE, fillEllipses = mclust.options("fillEllipses"), symbols = NULL, colors = NULL, scale = FALSE, xlim = NULL, ylim = NULL, cex = 1, PCH = ".", main = FALSE, ...)
```

**Arguments**

- `data` A numeric matrix or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- `dimens` A vector of length 2 giving the integer dimensions of the desired coordinate projections. The default is `c(1,2)`, in which the first dimension is plotted against the second.
- `parameters` A named list giving the parameters of an `MCLUST` model, used to produce superimposing ellipses on the plot. The relevant components are as follows:
  - `mean` The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.
  - `variance` A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
- `z` A matrix in which the [i,k]th entry gives the probability of observation i belonging to the kth class. Used to compute classification and uncertainty if those arguments aren’t available.
- `classification` A numeric or character vector representing a classification of observations (rows) of `data`. If present argument `z` will be ignored.
- `truth` A numeric or character vector giving a known classification of each data point. If `classification` or `z` is also present, this is used for displaying classification errors.
- `uncertainty` A numeric vector of values in (0,1) giving the uncertainty of each data point. If present argument `z` will be ignored.
choose from one of the following three options: "classification" (default), "error", "uncertainty".

addEllipses A logical indicating whether or not to add ellipses with axes corresponding to
the within-cluster covariances in case of "classification" or "uncertainty" plots.

fillEllipses A logical specifying whether or not to fill ellipses with transparent colors when
addEllipses = TRUE.

symbols Either an integer or character vector assigning a plotting symbol to each unique
class in classification. Elements in colors correspond to classes in order of
appearance in the sequence of observations (the order used by the function
unique). The default is given by mclust.options("classPlotSymbols").

colors Either an integer or character vector assigning a color to each unique class in
classification. Elements in colors correspond to classes in order of appear-
cance in the sequence of observations (the order used by the function unique).
The default is given by mclust.options("classPlotColors").

scale A logical variable indicating whether or not the two chosen dimensions should
be plotted on the same scale, and thus preserve the shape of the distribution.
Default: scale=FALSE

xlim, ylim Arguments specifying bounds for the ordinate, abscissa of the plot. This may be
useful for when comparing plots.

cex A numerical value specifying the size of the plotting symbols. The default value
is 1.

PCH An argument specifying the symbol to be used when a classification has not
been specified for the data. The default value is a small dot ".".

main A logical variable or NULL indicating whether or not to add a title to the plot
identifying the dimensions used.

... Other graphics parameters.

Value
A plot showing a two-dimensional coordinate projection of the data, together with the location of
the mixture components, classification, uncertainty, and/or classification errors.

See Also
clPairs, randProj, mclust2Dplot, mclust.options

Examples
## Not run:
est <- meVVV(iris[, -5], unmap(iris[, 5]))
par(pty = "s", mfrow = c(1, 1))
coordProj(iris[, -5], dimens=c(2,3), parameters = est$parameters, z = est$z,
        what = "classification", main = TRUE)
coordProj(iris[, -5], dimens=c(2,3), parameters = est$parameters, z = est$z,
        truth = iris[, 5], what = "error", main = TRUE)
coordProj(iris[, -5], dimens=c(2,3), parameters = est$parameters, z = est$z,
Weighted means, covariance and scattering matrices conditioning on a weighted matrix

Description

Compute efficiently (via Fortran code) the means, covariance and scattering matrices conditioning on a weighted or indicator matrix

Usage

covw(X, Z, normalize = TRUE)

Arguments

X  A \((nxp)\) data matrix, with \(n\) observations on \(p\) variables.
Z  A \((nxG)\) matrix of weights, with \(G\) number of groups.
normalize  A logical indicating if rows of \(Z\) should be normalized to sum to one.

Value

A list with the following components:

mean  A \((pxG)\) matrix of weighted means.
S  A \((pxpxG)\) array of weighted covariance matrices.
W  A \((pxpxG)\) array of weighted scattering matrices.

Author(s)

M. Fop and L. Scrucca

Examples

# Z as an indicator matrix
X <- iris[,1:4]
Z <- unmap(iris$Species)
str(covw(X, Z))
# Z as a matrix of weights
mod <- Mclust(X, G = 3, modelNames = "VvV")
str(covw(X, mod$z))
cross  

Simulated Cross Data

description

A 500 by 3 matrix in which the first column is the classification and the remaining columns are two data from a simulation of two crossed elliptical Gaussians.

Usage

data(cross)

Examples

# This dataset was created as follows
## Not run:
  n <- 250
  set.seed(0)
  cross <- rbind(matrix(rnorm(n*2), n, 2) %*% diag(c(1,9)),
                  matrix(rnorm(n*2), n, 2) %*% diag(c(1,9))[2:1])
  cross <- cbind(c(rep(1,n),rep(2,n)), cross)
## End(Not run)

cvMclustDA  
MclustDA cross-validation

description

K-fold cross-validation for discriminant analysis based on Gaussian finite mixture modeling.

Usage

cvMclustDA(object, nfold = 10,
            metric = c("error", "brier"),
            prop = object$prop,
            verbose = interactive(), ...)

Arguments

object                An object of class 'MclustDA' resulting from a call to MclustDA.
nfold                 An integer specifying the number of folds.
metric                A character string specifying the statistic to be used in the cross-validation resampling process. Possible values are "error" for the classification error, and "brier" for the Brier score.
prop        A vector of class prior probabilities, which if not provided default to the class proportions in the training data.

verbose     A logical controlling if a text progress bar is displayed during the cross-validation procedure. By default is TRUE if the session is interactive, and FALSE otherwise.

...         Further arguments passed to or from other methods.

**Value**

The function returns a list with the following components:

- **classification** a factor of cross-validated class labels.
- **z** a matrix containing the cross-validated probabilities for class assignment.
- **error** the cross-validation classification error if metric = "error", NA otherwise.
- **brier** the cross-validation Brier score if metric = "brier", NA otherwise.
- **se** the standard error of the cross-validated statistic.

**Author(s)**

Luca Scrucca

**See Also**


**Examples**

```r
## Not run:
X <- iris[, -5]
Class <- iris[, 5]

# common EEE covariance structure (which is essentially equivalent to linear discriminant analysis)
irisMclustDA <- MclustDA(X, Class, modelType = "EDDA", modelNames = "EEE")
cv <- cvMclustDA(irisMclustDA) # default 10-fold CV
cv[c("error", "se")]

cv <- cvMclustDA(irisMclustDA, nfold = length(Class)) # LOO-CV
cv[c("error", "se")]

cv <- cvMclustDA(irisMclustDA, metric = "brier") # 10-fold CV with Brier score metric
cv[c("brier", "se")]

# general covariance structure selected by BIC
irisMclustDA <- MclustDA(X, Class)
cv <- cvMclustDA(irisMclustDA) # default 10-fold CV
cv[c("error", "se")]
cv <- cvMclustDA(irisMclustDA, metric = "brier") # 10-fold CV with Brier score metric
cv[c("brier", "se")]

## End(Not run)
```
decomp2sigma

Convert mixture component covariances to matrix form

Description
Converts covariances from a parameterization by eigenvalue decomposition or cholesky factorization to representation as a 3-D array.

Usage
decomp2sigma(d, G, scale, shape, orientation, ...)

Arguments
d The dimension of the data.
G The number of components in the mixture model.
scale Either a G-vector giving the scale of the covariance (the dth root of its determinant) for each component in the mixture model, or a single numeric value if the scale is the same for each component.
shape Either a G by d matrix in which the kth column is the shape of the covariance matrix (normalized to have determinant 1) for the kth component, or a d-vector giving a common shape for all components.
orientation Either a d by d by G array whose [,,k]th entry is the orthonomal matrix whose columns are the eigenvectors of the covariance matrix of the kth component, or a d by d orthonormal matrix if the mixture components have a common orientation. The orientation component of decomp can be omitted in spherical and diagonal models, for which the principal components are parallel to the coordinate axes so that the orientation matrix is the identity.
...
Catches unused arguments from an indirect or list call via do.call.

Value
A 3-D array whose [,,k]th component is the covariance matrix of the kth component in an MVN mixture model.

See Also
sigma2decomp

Examples
meEst <- meVEV(iris[,-5], unmap(iris[,5]))
names(meEst)
meEst$parameters$variance
de <- meEst$parameters$variance
defaultPrior

Default conjugate prior for Gaussian mixtures

Description

Default conjugate prior specification for Gaussian mixtures.

Usage

defaultPrior(data, G, modelName, ...)

Arguments

data A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

G The number of mixture components.

modelName A character string indicating the model:

"E": equal variance (univariate)
"V": variable variance (univariate)
"EII": spherical, equal volume
"VII": spherical, unequal volume
"EEI": diagonal, equal volume and shape
"VEI": diagonal, varying volume, equal shape
"EVI": diagonal, equal volume, varying shape
"VVI": diagonal, varying volume and shape
"EEE": ellipsoidal, equal volume, shape, and orientation
"EEV": ellipsoidal, equal volume and equal shape
"VEV": ellipsoidal, equal shape
"VVV": ellipsoidal, varying volume, shape, and orientation.

A description of the models above is provided in the help of mclustModelNames. Note that in the multivariate case only 10 out of 14 models may be used in conjunction with a prior, i.e. those available in MCLUST up to version 4.4.

... One or more of the following:

dof The degrees of freedom for the prior on the variance. The default is d + 2, where d is the dimension of the data.

scale The scale parameter for the prior on the variance. The default is var(data)/G*(2/d), where d is the dimension of the data.
shrinkage The shrinkage parameter for the prior on the mean. The default value is 0.01. If 0 or NA, no prior is assumed for the mean.

mean The mean parameter for the prior. The default value is \( \text{colMeans(data)} \).

Details
defaultPrior is a function whose default is to output the default prior specification for EM within \textit{MCLUST}.
Furthermore, \texttt{defaultPrior} can be used as a template to specify alternative parameters for a conjugate prior.

Value
A list giving the prior degrees of freedom, scale, shrinkage, and mean.

References

See Also
\texttt{mclustBIC}, \texttt{me}, \texttt{mstep}, \texttt{priorControl}

Examples
\begin{verbatim}
# default prior
irisBIC <- mclustBIC(iris[,1:4], prior = priorControl())
summary(irisBIC, iris[,1:4])

# equivalent to previous example
irisBIC <- mclustBIC(iris[,1:4],
  prior = priorControl(functionName = "defaultPrior"))
summary(irisBIC, iris[,1:4])

# no prior on the mean; default prior on variance
irisBIC <- mclustBIC(iris[,1:4], prior = priorControl(shrinkage = 0))
summary(irisBIC, iris[,1:4])

# equivalent to previous example
irisBIC <- mclustBIC(iris[,1:4], prior =
    priorControl(functionName="defaultPrior", shrinkage=0))
summary(irisBIC, iris[,1:4])

defaultPrior( iris[,1:4], G = 3, modelName = "VVV")
\end{verbatim}
density for parameterized MVN mixtures

Description

Computes densities of observations in parameterized MVN mixtures.

Usage

dens(modelName, data, logarithm = FALSE, parameters, warn=NULL, ...)

Arguments

modelName : A character string indicating the model. The help file for mclustModelNames describes the available models.
data : A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
logarithm : A logical value indicating whether or not the logarithm of the component densities should be returned. The default is to return the component densities, obtained from the log component densities by exponentiation.
parameters : The parameters of the model:
  pro : The vector of mixing proportions for the components of the mixture.
  mean : The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.
  variance : A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.
  warn : A logical value indicating whether or not a warning should be issued when computations fail. The default is warn=FALSE.
...
Catches unused arguments in indirect or list calls via do.call.

Value

A numeric vector whose ith component is the density of the ith observation in data in the MVN mixture specified by parameters.

See Also

cdens, mclust.options, do.call
Examples

```r
## Not run:
faithfulModel <- Mclust(faithful)
Dens <- dens(modelName = faithfulModel$modelName, data = faithful,
parameters = faithfulModel$parameters)
Dens

## alternative call
do.call("dens", faithfulModel)
## End(Not run)
```

densityMclust  

Density Estimation via Model-Based Clustering

Description

Produces a density estimate for each data point using a Gaussian finite mixture model from Mclust.

Usage

densityMclust(data, ...)

Arguments

- **data**: A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- **...**: Additional arguments for the Mclust function. In particular, setting the arguments `G` and `modelNames` allow to specify the number of mixture components and the type of model to be fitted. By default an "optimal" model is selected based on the BIC criterion.

Value

An object of class densityMclust, which inherits from Mclust, is returned with the following slot added:

- **density**: The density evaluated at the input data computed from the estimated model.

Author(s)

Revised version by Luca Scrucca based on the original code by C. Fraley and A.E. Raftery.
References


See Also


Examples

```r
dens <- densityMclust(faithful$waiting)
summary(dens)
summary(dens, parameters = TRUE)
plot(dens, what = "BIC", legendArgs = list(x = "topright"))
plot(dens, what = "density", data = faithful$waiting)

dens <- densityMclust(faithful, modelNames = "EEE", G = 3)
summary(dens)
summary(dens, parameters = TRUE)
plot(dens, what = "density", data = faithful, 
      drawlabels = FALSE, points.pch = 20)
plot(dens, what = "density", type = "hdr")
plot(dens, what = "density", type = "hdr", prob = c(0.1, 0.9))
plot(dens, what = "density", type = "hdr", data = faithful)
plot(dens, what = "density", type = "persp")

## Not run:
dens <- densityMclust(iris[,1:4], G = 2)
summary(dens, parameters = TRUE)
plot(dens, what = "density", data = iris[,1:4], 
     col = "slategrey", drawlabels = FALSE, nlevels = 7)
plot(dens, what = "density", type = "hdr", data = iris[,1:4])
plot(dens, what = "density", type = "persp", col = grey(0.9))

## End(Not run)
```

densityMclust.diagnostic

*Diagnostic plots for mclustDensity estimation*

Description

Diagnostic plots for density estimation. Only available for the one-dimensional case.
densityMclust.diagnostics

Usage

densityMclust.diagnostics(object, type = c("cdf", "qq"),
col = c("black", "black"),
lwd = c(2,1), lty = c(1,1),
legend = TRUE, grid = TRUE,
...)

Arguments

object An object of class 'mclustDensity' obtained from a call to densityMclust function.
type The type of graph requested:
"cdf" = a plot of the estimated CDF versus the empirical distribution function.
"qq" = a Q-Q plot of sample quantiles versus the quantiles obtained from the inverse of the estimated cdf.
col A pair of values for the color to be used for plotting, respectively, the estimated CDF and the empirical cdf.
lwd A pair of values for the line width to be used for plotting, respectively, the estimated CDF and the empirical cdf.
lty A pair of values for the line type to be used for plotting, respectively, the estimated CDF and the empirical cdf.
legend A logical indicating if a legend must be added to the plot of fitted CDF vs the empirical CDF.
grid A logical indicating if a grid should be added to the plot.
... Additional arguments.

details

The two diagnostic plots for density estimation in the one-dimensional case are discussed in Loader (1999, pp- 87-90).

Author(s)

Luca Scrucca

References


See Also
densityMclust.plot.densityMclust.
Examples

```r
## Not run:
x <- faithful$waiting
dens <- densityMclust(x)
plot(dens, x, what = "diagnostic")
# or
densityMclust.diagnostic(dens, type = "cdf")
densityMclust.diagnostic(dens, type = "qq")
## End(Not run)
```

diabetes

### Diabetes data

**Description**

The data set contains three measurements made on 145 non-obese adult patients classified into three groups.

**Usage**

```r
data(diabetes)
```

**Format**

A data frame with the following variables:

- **class** The type of diabetes: Normal, Overt, and Chemical.
- **glucose** Area under plasma glucose curve after a three hour oral glucose tolerance test (OGTT).
- **insulin** Area under plasma insulin curve after a three hour oral glucose tolerance test (OGTT).
- **sspg** Steady state plasma glucose.

**Source**

dmvnorm

Density of multivariate Gaussian distribution

Description

Efficiently computes the density of observations for a generic multivariate Gaussian distribution.

Usage

```r
dmvnorm(data, mean, sigma, log = FALSE)
```

Arguments

- `data`: A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- `mean`: A vector of means for each variable.
- `sigma`: A positive definite covariance matrix.
- `log`: A logical value indicating whether or not the logarithm of the densities should be returned.

Value

A numeric vector whose \( i \)th element gives the density of the \( i \)th observation in `data` for the multivariate Gaussian distribution with parameters `mean` and `sigma`.

See Also

`dnorm`, `dens`

Examples

```r
# univariate
ngrid <- 101
x <- seq(-5, 5, length = ngrid)
dens <- dmvnorm(x, mean = 1, sigma = 5)
plot(x, dens, type = "l")

# bivariate
ngrid <- 101
x1 <- x2 <- seq(-5, 5, length = ngrid)
mu <- c(1,0)
sigma <- matrix(c(1,0.5,0.5,2), 2, 2)
dens <- dmvnorm(as.matrix(expand.grid(x1, x2)), mu, sigma)
dens <- matrix(dens, ngrid, ngrid)
image(x1, x2, dens)
contour(x1, x2, dens, add = TRUE)
```
**Description**

Implements the EM algorithm for parameterized Gaussian mixture models, starting with the expectation step.

**Usage**

```r
em(data, modelName, parameters, prior ==NULL, control = emControl(),
    warn = NULL, ...)
```

**Arguments**

- **data**
  A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

- **modelName**
  A character string indicating the model. The help file for `mclustModelNames` describes the available models.

- **parameters**
  A names list giving the parameters of the model. The components are as follows:
  - `pro` Mixing proportions for the components of the mixture. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.
  - `mean` The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.
  - `variance` A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
  - `Vinv` An estimate of the reciprocal hypervolume of the data region. If set to NULL or a negative value, the default is determined by applying function `hypvol` to the data. Used only when `pro` includes an additional mixing proportion for a noise component.

- **prior**
  Specification of a conjugate prior on the means and variances. The default assumes no prior.

- **control**
  A list of control parameters for EM. The defaults are set by the call `emControl()`.

- **warn**
  A logical value indicating whether or not a warning should be issued when computations fail. The default is `warn=FALSE`.

- **...**
  Catches unused arguments in indirect or list calls via `do.call`.
Value

A list including the following components:

- **modelName** A character string identifying the model (same as the input argument).
- **n** The number of observations in the data.
- **d** The dimension of the data.
- **G** The number of mixture components.
- **z** A matrix whose \([i,k]\)th entry is the conditional probability of the \(i\)th observation belonging to the \(k\)th component of the mixture.
- **parameters** A vector whose \(k\)th component is the mixing proportion for the \(k\)th component of the mixture model. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.
- **mean** The mean for each component. If there is more than one component, this is a matrix whose \(k\)th column is the mean of the \(k\)th component of the mixture model.
- **variance** A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
- **Vinv** The estimate of the reciprocal hypervolume of the data region used in the computation when the input indicates the addition of a noise component to the model.
- **loglik** The log likelihood for the data in the mixture model.
- **control** The list of control parameters for EM used.
- **prior** The specification of a conjugate prior on the means and variances used, NULL if no prior is used.

Attributes:

- "info" Information on the iteration.
- "WARNING" An appropriate warning if problems are encountered in the computations.

See Also

emE..., emVVV, estep, me, mstep, mclust.options, do.call

Examples

```r
## Not run:
msEst <- mstep(modelName = "EEE", data = iris[, -5],
               z = unmap(iris[,5]))
names(msEst)
em(modelName = msEst$modelName, data = iris[, -5],
    parameters = msEst$parameters)
do.call("em", c(list(data = iris[, -5]), msEst))  ## alternative call
```

## End(Not run)
emControl

Set control values for use with the EM algorithm

Description

Supplies a list of values including tolerances for singularity and convergence assessment, for use functions involving EM within MCLUST.

Usage

emControl(eps, tol, itmax, equalPro)

Arguments

deps

A scalar tolerance associated with deciding when to terminate computations due to computational singularity in covariances. Smaller values of eps allow computations to proceed nearer to singularity. The default is the relative machine precision .Machine$double.eps, which is approximately $2 \epsilon = 16 \times 10^3$ on IEEE-compliant machines.

tol

A vector of length two giving relative convergence tolerances for the log-likelihood and for parameter convergence in the inner loop for models with iterative M-step ("VEI", "VEE", "EVE", "VVE", "VEV"), respectively. The default is c(1.e-5, sqrt(.Machine$double.eps)). If only one number is supplied, it is used as the tolerance for the outer iterations and the tolerance for the inner iterations is as in the default.

itmax

A vector of length two giving integer limits on the number of EM iterations and on the number of iterations in the inner loop for models with iterative M-step ("VEI", "VEE", "EVE", "VVE", "VEV"), respectively. The default is c(.Machine$integer.max, .Machine$integer.max) allowing termination to be completely governed by tol. If only one number is supplied, it is used as the iteration limit for the outer iteration only.

equalPro

Logical variable indicating whether or not the mixing proportions are equal in the model. Default: equalPro = FALSE.

Details

emControl is provided for assigning values and defaults for EM within MCLUST.

Value

A named list in which the names are the names of the arguments and the values are the values supplied to the arguments.

See Also

em, estep, me, mstep, mclustBIC
Examples

irisBIC <- mclustBIC(iris[, -5], control = emControl(tol = 1.e-6))
summary(irisBIC, iris[, -5])

Description

Implements the EM algorithm for a parameterized Gaussian mixture model, starting with the expectation step.

Usage

emE(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emV(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emX(data, prior = NULL, warn = NULL, ...)
emEII(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emVII(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emEEI(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emVEI(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emEVI(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emVVI(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emEEE(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emVEE(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emEVE(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emVVE(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emEEV(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emVEV(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emEVV(data, parameters, prior = NULL, control = emControl(), warn = NULL, ...)
emXII(data, prior = NULL, warn = NULL, ...)
emXXI(data, prior = NULL, warn = NULL, ...)
emXXX(data, prior = NULL, warn = NULL, ...)

Arguments

data A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

parameters The parameters of the model:

pro Mixing proportions for the components of the mixture. There should one more mixing proportion than the number of Gaussian components if the mixture model includes a Poisson noise term.
The mean for each component. If there is more than one component, this is a matrix whose \( k \)th column is the mean of the \( k \)th component of the mixture model.

**variance**  A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.

**Vinv**  An estimate of the reciprocal hypervolume of the data region. The default is determined by applying function `hypvol` to the data. Used only when \( \text{pro} \) includes an additional mixing proportion for a noise component.

**prior**  The default assumes no prior, but this argument allows specification of a conjugate prior on the means and variances through the function `priorControl`.

**control**  A list of control parameters for EM. The defaults are set by the call `emControl()`.

**warn**  A logical value indicating whether or not a warning should be issued whenever a singularity is encountered. The default is given in `mclust.options("warn")`.

...  Catches unused arguments in indirect or list calls via `do.call`.

### Value
A list including the following components:

- **modelName**  A character string identifying the model (same as the input argument).
- **z**  A matrix whose \([i,k]\)th entry is the conditional probability of the \(i\)th observation belonging to the \(k\)th component of the mixture.
- **parameters**  \( \text{pro} \)  A vector whose \( k \)th component is the mixing proportion for the \( k \)th component of the mixture model. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.
- **mean**  The mean for each component. If there is more than one component, this is a matrix whose \( k \)th column is the mean of the \( k \)th component of the mixture model.
- **variance**  A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
- **Vinv**  The estimate of the reciprocal hypervolume of the data region used in the computation when the input indicates the addition of a noise component to the model.
- **loglik**  The log likelihood for the data in the mixture model.

### Attributes
- "info"  Information on the iteration.
- "WARNING"  An appropriate warning if problems are encountered in the computations.

### See Also
- `me`, `mstep`, `mclustVariance`, `mclust.options`.
Examples

```r
## Not run:
msEst <- mstepEEE(data = iris[-5], z = unmap(iris[,5]))
names(msEst)
emEEE(data = iris[-5], parameters = msEst$parameters)
## End(Not run)
```

---

**entPlot**

*Plot Entropy Plots*

**Description**

Plot "entropy plots" to help select the number of classes from a hierarchy of combined clusterings.

**Usage**

```r
entPlot(z, combiM, abc = c("standard", "normalized"), reg = 2, ...)
```

**Arguments**

- `z`: A matrix whose \([i,k]\)th entry is the probability that observation \(i\) in the data belongs to the \(k\)th class, for the initial solution (ie before any combining). Typically, the one returned by `Mclust/BIC`.
- `combiM`: A list of "combining matrices" (as provided by `clustCombi`), ie \(\text{combiM}[[K]]\) is the matrix whose \(k\)th row contains only zeros, but in columns corresponding to the labels of the classes in the \((K+1)\)-classes solution to be merged to get the \(K\)-classes combined solution. `combiM` must contain matrices from \(K = \text{number of classes in } z\) to one.
- `abc`: Choose one or more of: "standard", "normalized", to specify whether the number of observations involved in each combining step should be taken into account to scale the plots or not.
- `reg`: The number of parts of the piecewise linear regression for the entropy plots. Choose one or more of: 2 (for 1 change-point), 3 (for 2 change-points).
- `...`: Other graphical arguments to be passed to the plot functions.

**Details**

Please see the article cited in the references for more details. A clear elbow in the "entropy plot" should suggest the user to consider the corresponding number(s) of class(es).
Value

if \(abc = \text{"standard"}\), plots the entropy against the number of clusters and the difference between the entropy of successive combined solutions against the number of clusters. if \(abc = \text{"normalized"}\), plots the entropy against the cumulated number of observations involved in the successive combining steps and the difference between the entropy of successive combined solutions divided by the number of observations involved in the corresponding combining step against the number of clusters.

Author(s)

J.-P. Baudry, A. E. Raftery, L. Scrucca

References


See Also

plot.clustCombi, combiPlot, clustCombi

Examples

## Not run:
data(Baudry_etal_2010_JCGS_examples)
# run Mclust to get the MclustOutput
output <- clustCombi(data = ex4.2, modelNames = "VII")

entPlot(output$MclustOutput$z, output$combiM, reg = c(2,3))
# legend: in red, the single-change-point piecewise linear regression;
# in blue, the two-change-point piecewise linear regression.

## End(Not run)

---

errorBars

**Draw error bars on a plot**

Description

Draw error bars at x from upper to lower. If \(\text{horizontal} = \text{FALSE}\) (default) bars are drawn vertically, otherwise horizontally.

Usage

\[
\text{errorBars}(x, \text{upper}, \text{lower}, \text{width} = 0.1, \text{code} = 3, \text{angle} = 90, \text{horizontal} = \text{FALSE}, \ldots)
\]
estep

E-step for parameterized Gaussian mixture models.

Description

Implements the expectation step of EM algorithm for parameterized Gaussian mixture models.

Usage

estep(data, modelName, parameters, warn = NULL, ...)

Arguments

- **x**: A vector of values where the bars must be drawn.
- **upper**: A vector of upper values where the bars must end.
- **lower**: A vector of lower values where the bars must start.
- **width**: A value specifying the width of the end-point segment.
- **code**: An integer code specifying the kind of arrows to be drawn. For details see `arrows`.
- **angle**: A value specifying the angle at the arrow edge. For details see `arrows`.
- **horizontal**: A logical specifying if bars should be drawn vertically (default) or horizontally.
- **...**: Further arguments are passed to `arrows`.

Examples

```r
par(mfrow=c(2,2))
# Create a simple example dataset
x <- 1:5
n <- c(10, 15, 12, 6, 3)
se <- c(1, 1.2, 2, 1, .5)
# upper and lower bars
b <- barplot(n, ylim = c(0, max(n)*1.5))
errorBars(b, lower = n-se, upper = n+se, lwd = 2, col = "red3")
# one side bars
b <- barplot(n, ylim = c(0, max(n)*1.5))
errorBars(b, lower = n, upper = n+se, lwd = 2, col = "red3", code = 1)
# plot(x, n, ylim = c(0, max(n)*1.5), pch = 0)
errorBars(x, lower = n-se, upper = n+se, lwd = 2, col = "red3")
# dotchart(n, labels = x, pch = 19, xlim = c(0, max(n)*1.5))
errorBars(x, lower = n-se, upper = n+se, col = "red3", horizontal = TRUE)
```
Arguments

data A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

modelName A character string indicating the model. The help file for mclustModelNames describes the available models.

parameters A names list giving the parameters of the model. The components are as follows:

pro Mixing proportions for the components of the mixture. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.

mean The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.

variance A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.

Vinv An estimate of the reciprocal hypervolume of the data region. If set to NULL or a negative value, the default is determined by applying function hypvol to the data. Used only when pro includes an additional mixing proportion for a noise component.

warn A logical value indicating whether or not a warning should be issued when computations fail. The default is warn=FALSE.

... Catches unused arguments in indirect or list calls via do.call.

Value

A list including the following components:

modelName A character string identifying the model (same as the input argument).

z A matrix whose [i,k]th entry is the conditional probability of the ith observation belonging to the kth component of the mixture.

parameters The input parameters.

loglik The log-likelihood for the data in the mixture model.

Attributes "WARNING": an appropriate warning if problems are encountered in the computations.

See Also
estepE, estepVVV, em, mstep, mclust.options, mclustVariance

Examples

## Not run:
msEst <- mstep(modelName = "VVV", data = iris[, -5], z = unmap(iris[,5]))
names(msEst)
estepE

estep(modelName = msEst$modelName, data = iris[-5],
parameters = msEst$parameters)
## End(Not run)

---

estepE  

_E-step in the EM algorithm for a parameterized Gaussian mixture model._

**Description**

Implements the expectation step in the EM algorithm for a parameterized Gaussian mixture model.

**Usage**

```r
estepE(data, parameters, warn = NULL, ...)
estepV(data, parameters, warn = NULL, ...)
estepEII(data, parameters, warn = NULL, ...)
estepVII(data, parameters, warn = NULL, ...)
estepEEI(data, parameters, warn = NULL, ...)
estepVEI(data, parameters, warn = NULL, ...)
estepEVI(data, parameters, warn = NULL, ...)
estepVVI(data, parameters, warn = NULL, ...)
estepEEE(data, parameters, warn = NULL, ...)
estepEEV(data, parameters, warn = NULL, ...)
estepVEV(data, parameters, warn = NULL, ...)
estepVVE(data, parameters, warn = NULL, ...)
estepVVE(data, parameters, warn = NULL, ...)
```

**Arguments**

- **data**
  
  A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

- **parameters**
  
  The parameters of the model:
  
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pro</td>
<td>Mixing proportions for the components of the mixture. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.</td>
</tr>
<tr>
<td>mu</td>
<td>The mean for each component. If there is more than one component, this is a matrix whose columns are the means of the components.</td>
</tr>
<tr>
<td>variance</td>
<td>A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for <code>mclustVariance</code> for details.</td>
</tr>
</tbody>
</table>
Vinv  An estimate of the reciprocal hypervolume of the data region. If not supplied or set to a negative value, the default is determined by applying function \texttt{hypvol} to the data. Used only when \texttt{pro} includes an additional mixing proportion for a noise component.

\texttt{warn}  A logical value indicating whether or certain warnings should be issued. The default is given by \texttt{mclust.options("warn").}

\texttt{...}  Catches unused arguments in indirect or list calls via \texttt{do.call}.

\textbf{Value}

A list including the following components:

\begin{itemize}
  \item \texttt{modelName}  Character string identifying the model.
  \item \texttt{z}  A matrix whose \([i,k]\)th entry is the conditional probability of the \(i\)th observation belonging to the \(k\)th component of the mixture.
  \item \texttt{parameters}  The input parameters.
  \item \texttt{loglik}  The loglikelihood for the data in the mixture model.
\end{itemize}

\item \texttt{Attribute}  "\texttt{WARNING}": An appropriate warning if problems are encountered in the computations.

\textbf{See Also}

\texttt{estep, em, mstep, do.call, mclustVariance, mclust.options}.

\textbf{Examples}

\begin{verbatim}
## Not run:
msEst <- mstepEII(data = iris[,-5], z = unmap(iris[,5]))
names(msEst)
estepEII(data = iris[,-5], parameters = msEst$parameters)
## End(Not run)
\end{verbatim}

\textbf{EuroUnemployment}  \textit{Unemployment data for European countries in 2014}

\textbf{Description}

The data set contains unemployment rates for 31 European countries for the year 2014.

\textbf{Usage}

\texttt{data(EuroUnemployment)}
Format

A data frame with the following variables:

TUR  Total unemployment rate, i.e. percentage of unemployed persons aged 15-74 in the economically active population.

YUR  Youth unemployment rate, i.e. percentage of unemployed persons aged 15-24 in the economically active population.

LUR  Long-term unemployment rate, i.e. percentage of unemployed persons who have been unemployed for 12 months or more.

Source

EUROSTAT (http://ec.europa.eu/eurostat/web/lfs/data/database)

Description

Starting with the density estimate obtained from a fitted Gaussian finite mixture model, cluster cores are identified from the connected components at a given density level. Once cluster cores are identified, the remaining observations are allocated to those cluster cores for which the probability of cluster membership is the highest.

Usage

```r
gmmhd(object,
     ngrid = min(round((log(nrow(data)))*10), nrow(data)),
     dr = list(d = 3, lambda = 1, cumEvalues = NULL, mindir = 2),
     classify = list(G = 1:5,
                     modelNames = mclust.options("emModelNames")[-c(8, 10)],
                     ...)
)
```

## S3 method for class 'gmmhd'

```r
plot(x, what = c("mode", "cores", "clusters"), ...)
```

Arguments

- **object**: An object returned by `Mclust`.
- **ngrid**: An integer specifying the number of grid points used to compute the density levels.
- **dr**: A list of parameters used in the dimension reduction step.
- **classify**: A list of parameters used in the classification step.
- **x**: An object of class 'gmmhd' as returned by the function `gmmhd`.
- **what**: A string specifying the type of plot to be produced. See Examples section.
- **...**: further arguments passed to or from other methods.
Details

Model-based clustering associates each component of a finite mixture distribution to a group or cluster. An underlying implicit assumption is that a one-to-one correspondence exists between mixture components and clusters. However, a single Gaussian density may not be sufficient, and two or more mixture components could be needed to reasonably approximate the distribution within a homogeneous group of observations.

This function implements the methodology proposed by Scrucca (2016) based on the identification of high density regions of the underlying density function. Starting with an estimated Gaussian finite mixture model, the corresponding density estimate is used to identify the cluster cores, i.e. those data points which form the core of the clusters. These cluster cores are obtained from the connected components at a given density level $c$. A mode function gives the number of connected components as the level $c$ is varied. Once cluster cores are identified, the remaining observations are allocated to those cluster cores for which the probability of cluster membership is the highest.

The method usually improves the identification of non-Gaussian clusters compared to a fully parametric approach. Furthermore, it enables the identification of clusters which cannot be obtained by merging mixture components, and it can be straightforwardly extended to cases of higher dimensionality.

Value

A list of class gmmhd with the following components:

- **Mclust**: The input object of class "Mclust" representing an estimated Gaussian finite mixture model.
- **MclustDA**: An object of class "MclustDA" containing the model used for the classification step.
- **MclustDR**: An object of class "MclustDR" containing the dimension reduction step if performed, otherwise NULL.
- **x**: The data used in the algorithm. This can be the input data or a projection if a preliminary dimension reduction step is performed.
- **density**: The density estimated from the input Gaussian finite mixture model evaluated at the input data.
- **con**: A list of connected components at each step.
- **nc**: A vector giving the number of connected components (i.e. modes) at each step.
- **pn**: Vector of values over a uniform grid of proportions of length ngrid.
- **qn**: Vector of density quantiles corresponding to proportions pn.
- **pc**: Vector of empirical proportions corresponding to quantiles qn.
- **clusterCores**: Vector of cluster cores numerical labels; NAs indicate that an observation does not belong to any cluster core.
- **numClusters**: An integer giving the number of clusters.

Author(s)

Luca Scrucca <luca.scrucca@unipg.it>
**References**


**See Also**

Mclust

**Examples**

```r
## Not run:
data(faithful)
mod <- Mclust(faithful)
summary(mod)
plot(as.densityMclust(mod), faithful, what = "density",
     points.pch = mclust.options("classPlotSymbols")[mod$classification],
     points.col = mclust.options("classPlotColors")[mod$classification])

GMMHD <- gmmhd(mod)
summary(GMMHD)

plot(GMMHD, what = "mode")
plot(GMMHD, what = "cores")
plot(GMMHD, what = "clusters")
## End(Not run)
```

---

**GvHD**

<table>
<thead>
<tr>
<th><strong>GvHD Dataset</strong></th>
</tr>
</thead>
</table>

**Description**

GvHD (Graft-versus-Host Disease) data of Brinkman et al. (2007). Two samples of this flow cytometry data, one from a patient with the GvHD, and the other from a control patient. The GvHD positive and control samples consist of 9083 and 6809 observations, respectively. Both samples include four biomarker variables, namely, CD4, CD8b, CD3, and CD8. The objective of the analysis is to identify CD3+ CD4+ CD8b+ cell sub-populations present in the GvHD positive sample.

A treatment of this data by combining mixtures is proposed in Baudry et al. (2010).

**Usage**

data(GvHD)
Format

GvHD.pos (positive patient) is a data frame with 9083 observations on the following 4 variables, which are biomarker measurements.

CD4
CD8b
CD3
CD8

GvHD.control (control patient) is a data frame with 6809 observations on the following 4 variables, which are biomarker measurements.

CD4
CD8b
CD3
CD8

References


Examples

```r
## Not run:
data(GvHD)
dat <- GvHD.pos[1:500,] # only a few lines for a quick example
output <- clustCombi(data = dat)
output # is of class clustCombi
# plot the hierarchy of combined solutions
plot(output, what = "classification")
# plot some "entropy plots" which may help one to select the number of classes
plot(output, what = "entropy")
# plot the tree structure obtained from combining mixture components
plot(output, what = "tree")

## End(Not run)
```
Model-based Agglomerative Hierarchical Clustering

Description

Agglomerative hierarchical clustering based on maximum likelihood criteria for Gaussian mixture models parameterized by eigenvalue decomposition.

Usage

hc(data,
   modelName = mclust.options("hcModelName"),
   partition, minclus = 1, ...,
   use = mclust.options("hcUse"))

Arguments

data  A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations ($n$) and columns correspond to variables ($d$).

modelName  A character string indicating the model to be used. Possible models are:

"E" equal variance (one-dimensional)
"V" spherical, variable variance (one-dimensional)
"EII" spherical, equal volume
"VII" spherical, unequal volume
"EEE" ellipsoidal, equal volume, shape, and orientation
"VVV" ellipsoidal, varying volume, shape, and orientation.

By default the model provided by mclust.options("hcModelName") is used. See mclust.options.

partition  A numeric or character vector representing a partition of observations (rows) of data. If provided, group merges will start with this partition. Otherwise, each observation is assumed to be in a cluster by itself at the start of agglomeration.

minclus  A number indicating the number of clusters at which to stop the agglomeration. The default is to stop when all observations have been merged into a single cluster.

...  Arguments for the method-specific hc functions. See for example hcE.

use  A string or a vector of character strings specifying the type of input variables/data transformation to be used for model-based hierarchical clustering. By default the method specified in mclust.options("hcUse") is used. See mclust.options.
Details

Most models have memory usage of the order of the square of the number groups in the initial partition for fast execution. Some models, such as equal variance or "EEE", do not admit a fast algorithm under the usual agglomerative hierarchical clustering paradigm. These use less memory but are much slower to execute.

Value

The function \( hc() \) returns a numeric two-column matrix in which the \( i \)th row gives the minimum index for observations in each of the two clusters merged at the \( i \)th stage of agglomerative hierarchical clustering. Several other informations are also returned as attributes.

The plotting method \( \text{plot.hc}() \) draws a dendrogram, which can be based on either the classification loglikelihood or the merge level (number of clusters). For details, see the associated help file.

Note

If \( \text{modelName} = \text"E" \) (univariate with equal variances) or \( \text{modelName} = \text"EII" \) (multivariate with equal spherical covariances), then underlying model is the same as that for Ward’s method for hierarchical clustering.

References


See Also

\( \text{hcE}, \ldots, \text{hcVVV}, \text{plot.hc}, \text{hclass}, \text{mclust.options} \)

Examples

```r
hcTree <- hc(modelName = "VVV", data = iris[, -5])
hcTree
c1 <- hclass(hcTree, c(2, 3))
table(c1[, "2"])
table(c1[, "3"])

## Not run:
c1Pairs(iris[, -5], classification = c1[, "2"])
c1Pairs(iris[, -5], classification = c1[, "3"])
## End(Not run)
```
Model-based Hierarchical Clustering

Description

Agglomerative hierarchical clustering based on maximum likelihood for a Gaussian mixture model parameterized by eigenvalue decomposition.

Usage

hcE(data, partition, minclus=1, ...)  
hcV(data, partition, minclus = 1, alpha = 1, ...)  
hcEI(data, partition, minclus = 1, ...)  
hcVII(data, partition, minclus = 1, alpha = 1, ...)  
hcEEE(data, partition, minclus = 1, ...)  
hcVVV(data, partition, minclus = 1, alpha = 1, beta = 1, ...)  

Arguments

data A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

partition A numeric or character vector representing a partition of observations (rows) of data. If provided, group merges will start with this partition. Otherwise, each observation is assumed to be in a cluster by itself at the start of agglomeration.

minclus A number indicating the number of clusters at which to stop the agglomeration. The default is to stop when all observations have been merged into a single cluster.

alpha, beta Additional tuning parameters needed for initialization in some models. For details, see Fraley 1998. The defaults provided are usually adequate.

... Catch unused arguments from a do.call call.

Details

Most models have memory usage of the order of the square of the number groups in the initial partition for fast execution. Some models, such as equal variance or "EEE", do not admit a fast algorithm under the usual agglomerative hierarchical clustering paradigm. These use less memory but are much slower to execute.

Value

A numeric two-column matrix in which the i\(^{th}\) row gives the minimum index for observations in each of the two clusters merged at the i\(^{th}\) stage of agglomerative hierarchical clustering.
References


See Also

hc, hclass hcRandomPairs

Examples

```r
hcTree <- hcEII(data = iris[-5])
c1 <- hclass(hcTree,c(2,3))

## Not run:
par(pty = "s", mfrow = c(1,1))
clPairs(iris[-5],cl=c1,"2")
clPairs(iris[-5],cl=c1,"3")

par(mfrow = c(1,2))
dimens <- c(1,2)
coordProj(iris[-5], classification=c1,"2", dimens=dimens)
coordProj(iris[-5], classification=c1,"3", dimens=dimens)

## End(Not run)
```

hclass

Classifications from Hierarchical Agglomeration

Description

Determines the classifications corresponding to different numbers of groups given merge pairs from hierarchical agglomeration.

Usage

`hclass(hcPairs, G)`

Arguments

- **hcPairs**: A numeric two-column matrix in which the $i$th row gives the minimum index for observations in each of the two clusters merged at the $i$th stage of agglomerative hierarchical clustering.
- **G**: An integer or vector of integers giving the number of clusters for which the corresponding classifications are wanted.
Value

A matrix with \(\text{length}(G)\) columns, each column corresponding to a classification. Columns are indexed by the character representation of the integers in \(G\).

See Also

hc, hcE

Examples

```r
hcTree <- hc(modelName="VVV", data = iris[, -5])
c1 <- hclass(hcTree, c(2, 3))

## Not run:
par(pty = "s", mfrow = c(1, 1))
clPairs(iris[, -5], cl = c1[, "2"])
clPairs(iris[, -5], cl = c1[, "3"])
## End(Not run)
```

hcRandomPairs

Random hierarchical structure

Description

Create a hierarchical structure using a random hierarchical partition of the data.

Usage

```r
hcRandomPairs(data, seed = NULL, ...)
```

Arguments

data A numeric matrix or data frame of observations. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

seed Optional single value, interpreted as an integer, specifying the seed for random partition.

... Catches unused arguments in indirect or list calls via \texttt{do.call}.

Value

A numeric two-column matrix in which the \(i\)th row gives the minimum index for observations in each of the two clusters merged at the \(i\)th stage of a random agglomerative hierarchical clustering.

See Also

hc, hclass hcVVV
Examples

```r
data <- iris[,1:4]
randPairs <- hcRandomPairs(data)
str(randPairs)
# start model-based clustering from a random partition
mod <- Mclust(data, initialization = list(hcPairs = randPairs))
summary(mod)
```

- `hdrlevels` 
  **Highest Density Region (HDR) Levels**

Description

Compute the levels of Highest Density Regions (HDRs) for any density and probability levels.

Usage

```r
hdrlevels(density, prob)
```

Arguments

- `density` A vector of density values computed on a set of (observed) evaluation points.
- `prob` A vector of probability levels in the range \([0, 1]\).

Details

From Hyndman (1996), let \(f(x)\) be the density function of a random variable \(X\). Then the \(100(1 - \alpha)\)\% HDR is the subset \(R(f_\alpha)\) of the sample space of \(X\) such that

\[
R(f_\alpha) = \{x : f(x) \geq f_\alpha\}
\]

where \(f_\alpha\) is the largest constant such that \(Pr(X \in R(f_\alpha)) \geq 1 - \alpha\).

Value

The function returns a vector of density values corresponding to HDRs at given probability levels.

Author(s)

L. Scrucca

References


See Also

- `plot.densityMclust`
Examples

# Example: univariate Gaussian
x <- rnorm(1000)
f <- dnorm(x)
a <- c(0.5, 0.25, 0.1)
(f_a <- hdrlevels(f, prob = 1-a))

plot(x, f)
abline(h = f_a, lty = 2)
text(max(x), f_a, labels = paste0("f", a), pos = 3)

mean(f > f_a[1])
range(x[which(f > f_a[1])])
qnorm(1-a[1]/2)

mean(f > f_a[2])
range(x[which(f > f_a[2])])
qnorm(1-a[2]/2)

mean(f > f_a[3])
range(x[which(f > f_a[3])])
qnorm(1-a[3]/2)

# Example 2: univariate Gaussian mixture
set.seed(1)
cl <- sample(1:2, size = 1000, prob = c(0.7, 0.3), replace = TRUE)
x <- ifelse(cl == 1,
            rnorm(1000, mean = 0, sd = 1),
            rnorm(1000, mean = 4, sd = 1))
f <- 0.7*dnorm(x, mean = 0, sd = 1) + 0.3*dnorm(x, mean = 4, sd = 1)
a <- 0.25
(f_a <- hdrlevels(f, prob = 1-a))

plot(x, f)
abline(h = f_a, lty = 2)
text(max(x), f_a, labels = paste0("f", a), pos = 3)

mean(f > f_a)

# find the regions of HDR
ord <- order(x)
ord <- ord
x_a <- x[ord]
j <- which.max(diff(x_a))
region1 <- x_a[c(1,j)]
region2 <- x_a[c(j+1,length(x_a))]
plot(x, f, type = "l")
abline(h = f_a, lty = 2)
abline(v = region1, lty = 3, col = 2)
abline(v = region2, lty = 3, col = 3)
Aproximate Hypervolume for Multivariate Data

Description

Computes a simple approximation to the hypervolume of a multivariate data set.

Usage

```
hypvol(data, reciprocal=FALSE)
```

Arguments

- **data**: A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- **reciprocal**: A logical variable indicating whether or not the reciprocal hypervolume is desired rather than the hypervolume itself. The default is to return the hypervolume.

Value

Returns the minimum of the hypervolume computed from simple variable bounds and that computed from variable bounds of the principal component scores. Used for the default hypervolume parameter for the noise component when observations are designated as noise in `Mclust` and `mclustBIC`.

References


See Also

- `mclustBIC`

Examples

```
hypvol(iris[,,-5])
```
Description
Computes the ICL (Integrated Complete-data Likelihood) for criterion for a Gaussian Mixture Model fitted by `Mclust`.

Usage
icl(object, ...)

Arguments
object
An object of class 'Mclust' resulting from a call to `Mclust`.

Value
The ICL for the given input MCLUST model.

References

See Also
`Mclust`, `mclustBIC`, `mclustICL`, `bic`.

Examples
mod <- Mclust(iris[,1:4])
icl(mod)
**imputeData**

*Missing data imputation via the mix package*

Description

Imputes missing data using the mix package.

Usage

```r
imputeData(data, categorical = NULL, seed = NULL, verbose = interactive())
```

Arguments

- `data`: A numeric vector, matrix, or data frame of observations containing missing values. Categorical variables are allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- `categorical`: A logical vector whose `i`th entry is `TRUE` if the `i`th variable or column of `data` is to be interpreted as categorical and `FALSE` otherwise. The default is to assume that a variable is to be interpreted as categorical only if it is a factor.
- `seed`: A seed for the function `rngseed` that is used to initialize the random number generator in mix. By default, a seed is chosen uniformly in the interval (.Machine$integer.max/1024,.Machine$integer.max).
- `verbose`: A logical, if `TRUE` reports info about iterations of the algorithm.

Value

A dataset of the same dimensions as `data` with missing values filled in.

References


See Also

- `imputePairs`

Examples

```r
## Not run:
# Note that package 'mix' must be installed
data(stlouis, package = "mix")

# impute the continuous variables in the stlouis data
stlimp <- imputeData(stlouis[,-(1:3)])

# plot imputed values
imputePairs(stlouis[,-(1:3)], stlimp)

## End(Not run)
```
imputePairs

Pairwise Scatter Plots showing Missing Data Imputations

Description

Creates a scatter plot for each pair of variables in given data, allowing display of imputations for missing values in different colors and symbols than non missing values.

Usage

```r
imputePairs(data, dataImp,
             symbols = c(1,16), colors = c("black", "red"), labels,
             panel = points, ..., lower.panel = panel, upper.panel = panel,
             diag.panel = NULL, text.panel = textPanel, label.pos = 0.5 +
             has.diag/3, cex.labels = NULL, font.labels = 1, rowlattop = TRUE,
             gap = 0.2)
```

Arguments

data A numeric vector, matrix, or data frame of observations containing missing values. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

dataImp The dataset data with missing values imputed.
symbols Either an integer or character vector assigning plotting symbols to the nonmissing data and imputed values, respectively. The default is a closed circle for the nonmissing data and an open circle for the imputed values.

colors Either an integer or character vector assigning colors to the nonmissing data and imputed values, respectively. The default is black for the nonmissing data and red for the imputed values.

labels As in function `pairs`.
panel As in function `pairs`.
... As in function `pairs`.
lower.panel As in function `pairs`.
upper.panel As in function `pairs`.
diag.panel As in function `pairs`.
text.panel As in function `pairs`.
label.pos As in function `pairs`.
cex.labels As in function `pairs`.
font.labels As in function `pairs`.
rowlattop As in function `pairs`.
gap As in function `pairs`.  

Value
A pairs plot displaying the location of missing and nonmissing values.

References

See Also
pairs, imputeData

Examples
## Not run:
# Note that package 'mix' must be installed
data(stlouis, package = "mix")

# impute the continuous variables in the stlouis data
stlimp <- imputeData(stlouis[, -(1:3)])

# plot imputed values
imputePairs(stlouis[, -(1:3)], stlimp)
## End(Not run)

logLik.Mclust Log-Likelihood of a Mclust object

Description
Returns the log-likelihood for a 'Mclust' object.

Usage
## S3 method for class 'Mclust'
logLik(object, ...)

Arguments
object an object of class 'Mclust' resulting from a call to Mclust.
... further arguments passed to or from other methods.

Value
Returns an object of class 'logLik' with an element providing the maximized log-likelihood, and further arguments giving the number of (estimated) parameters in the model ("df") and the sample size ("nobs").
logLik.MclustDA

Author(s)
Luca Scrucca

See Also
Mclust.

Examples

## Not run:
irisMclust <- Mclust(iris[,1:4])
summary(irisMclust)
logLik(irisMclust)
## End(Not run)

logLik.MclustDA Log-Likelihood of a MclustDA object

Description
Returns the log-likelihood for a MclustDA object.

Usage

## S3 method for class 'MclustDA'
logLik(object, data, ...)

Arguments

object an object of class 'MclustDA' resulting from a call to MclustDA.
data the data for which the log-likelihood must be computed. If missing, the observed data from the 'MclustDA' object is used.
...
进一步 arguments passed to or from other methods.

Value

Returns an object of class 'logLik' with an element providing the maximized log-likelihood, and further arguments giving the number of (estimated) parameters in the model ("df") and the sample size ("nobs").

Author(s)
Luca Scrucca

See Also
MclustDA.
Examples

```r
## Not run:
irisMclustDA <- MclustDA(iris[,1:4], iris$Species)
summary(irisMclustDA)
logLik(irisMclustDA)

## End(Not run)
```

```
majorityVote

Description

A function to compute the majority vote (some would say plurality) label in a vector of labels, breaking ties at random.

Usage

```r
majorityVote(x)
```

Arguments

- `x`: A vector of values, either numerical or not.

Value

A list with the following components:

- `table`: A table of votes for each unique value of `x`.
- `ind`: An integer specifying which unique value of `x` corresponds to the majority vote.
- `majority`: A string specifying the majority vote label.

Author(s)

L. Scrucca

Examples

```r
majorityVote(x)
```
map

Classification given Probabilities

Description

Converts a matrix in which each row sums to 1 to an integer vector specifying for each row the column index of the maximum.

Usage

map(z, warn = mclust.options("warn"), ...)

Arguments

z

A matrix (for example a matrix of conditional probabilities in which each row sums to 1 as produced by the E-step of the EM algorithm).

warn

A logical variable indicating whether or not a warning should be issued when there are some columns of z for which no row attains a maximum.

...

Provided to allow lists with elements other than the arguments can be passed in indirect or list calls with do.call.

Value

A integer vector with one entry for each row of z, in which the i-th value is the column index at which the i-th row of z attains a maximum.

See Also

unmap, estep, em, me.

Examples

emEst <- me(modelName = "VVV", data = iris[, -5], z = unmap(iris[, 5]))

map(emEst$z)

mapClass

Correspondence between classifications

Description

Best correspondence between classes given two vectors viewed as alternative classifications of the same object.
Usage

mapClass(a, b)

Arguments

a  A numeric or character vector of class labels.

b  A numeric or character vector of class labels. Must have the same length as a.

Value

A list with two named elements, aTo b and bTo a which are themselves lists. The aTo b list has a component corresponding to each unique element of a, which gives the element or elements of b that result in the closest class correspondence.

The bTo a list has a component corresponding to each unique element of b, which gives the element or elements of a that result in the closest class correspondence.

See Also

classError, table

Examples

a <- rep(1:3, 3)
a
b <- rep(c("A", "B", "C"), 3)
b
mapClass(a, b)
a <- sample(1:3, 9, replace = TRUE)
a
b <- sample(c("A", "B", "C"), 9, replace = TRUE)
b
mapClass(a, b)

Mclust  Model-Based Clustering

description

Model-based clustering based on parameterized finite Gaussian mixture models. Models are estimated by EM algorithm initialized by hierarchical model-based agglomerative clustering. The optimal model is then selected according to BIC.
Usage

Mclust(data, G = NULL, modelNames = NULL,
      prior = NULL,
      control = emControl(),
      initialization = NULL,
      warn = mclust.options("warn"),
      x = NULL,
      verbose = interactive(), ...)

Arguments

data A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations (n) and columns correspond to variables (d).

G An integer vector specifying the numbers of mixture components (clusters) for which the BIC is to be calculated. The default is G=1:9.

modelNames A vector of character strings indicating the models to be fitted in the EM phase of clustering. The default is:
   • for univariate data (d = 1): c("E","V")
   • for multivariate data (n > d): all the models available in mclust.options("emModelNames")
   • for multivariate data (n <= d): the spherical and diagonal models, i.e. c("EII","VII","EEI","EVI","VEI","VVI")

The help file for mclustModelNames describes the available models.

prior The default assumes no prior, but this argument allows specification of a conjugate prior on the means and variances through the function priorControl. Note that, as described in defaultPrior, in the multivariate case only 10 out of 14 models may be used in conjunction with a prior, i.e. those available in MCLUST up to version 4.4.

control A list of control parameters for EM. The defaults are set by the call emControl().

initialization A list containing zero or more of the following components:

hcPairs A matrix of merge pairs for hierarchical clustering such as produced by function hc.

For multivariate data, the default is to compute a hierarchical agglomerative clustering tree by applying function hc with model specified by mclust.options("hcmodelName"), and data transformation set by mclust.options("hcUse").

All the input or a subset as indicated by the subset argument is used for initial clustering.

The hierarchical clustering results are then used to start the EM algorithm from a given partition.

For univariate data, the default is to use quantiles to start the EM algorithm. However, hierarchical clustering could also be used by calling hc with model specified as "V" or "E".

subset A logical or numeric vector specifying a subset of the data to be used in the initial hierarchical clustering phase. By default no subset is used unless the number of observations exceeds the value specified by mclust.options("subset").
Note that to guarantee exact reproducibility of results a seed must be specified (see `set.seed`).

*noise* A logical or numeric vector indicating an initial guess as to which observations are noise in the data. If numeric the entries should correspond to row indexes of the data. If supplied, a noise term will be added to the model in the estimation.

*warn* A logical value indicating whether or not certain warnings (usually related to singularity) should be issued. The default is controlled by `mclust.options`.

*x* An object of class `mclustBIC`. If supplied, BIC values for models that have already been computed and are available in `x` are not recomputed. All arguments, with the exception of `data`, `G` and `modelName`, are ignored and their values are set as specified in the attributes of `x`. Defaults for `G` and `modelNames` are taken from `x`.

*verbose* A logical controlling if a text progress bar is displayed during the fitting procedure. By default is `TRUE` if the session is interactive, and `FALSE` otherwise.

*...* Catches unused arguments in indirect or list calls via `do.call`.

**Value**

An object of class `Mclust` providing the optimal (according to BIC) mixture model estimation. The details of the output components are as follows:

*call* The matched call

*data* The input data matrix.

*modelName* A character string denoting the model at which the optimal BIC occurs.

*n* The number of observations in the data.

*d* The dimension of the data.

*G* The optimal number of mixture components.

*BIC* All BIC values.

*loglik* The log-likelihood corresponding to the optimal BIC.

*df* The number of estimated parameters.

*bic* BIC value of the selected model.

*icl* ICL value of the selected model.

*hypvol* The hypervolume parameter for the noise component if required, otherwise set to NULL (see `hypvol`).

*parameters* A list with the following components:

*pro* A vector whose kth component is the mixing proportion for the kth component of the mixture model. If missing, equal proportions are assumed.

*mean* The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.

*variance* A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
A matrix whose \( [i,k] \)th entry is the probability that observation \( i \) in the test data belongs to the \( k \)th class.

**classification**  The classification corresponding to \( z \), i.e. \( \text{map}(z) \).

**uncertainty**  The uncertainty associated with the classification.

### References


### See Also

`summary.Mclust`, `plot.Mclust`, `priorControl`, `emControl`, `hc`, `mclustBIC`, `mclustModelNames`, `mclust.options`

### Examples

```r
mod1 <- Mclust(iris[,1:4])
summary(mod1)

mod2 <- Mclust(iris[,1:4], G = 3)
summary(mod2, parameters = TRUE)

# Using prior
mod3 <- Mclust(iris[,1:4], prior = priorControl())
summary(mod3)

mod4 <- Mclust(iris[,1:4], prior = priorControl(functionName="defaultPrior", shrinkage=0.1))
summary(mod4)

# Clustering of faithful data with some artificial noise added
nNoise <- 100
set.seed(0) # to make it reproducible
Noise <- apply(faithful, 2, function(x)
               runif(nNoise, min = min(x)-.1, max = max(x)+.1))
data <- rbind(faithful, Noise)
plot(faithful)
points(Noise, pch = 20, cex = 0.5, col = "lightgrey")
set.seed(0)
NoiseInit <- sample(c(TRUE,FALSE), size = nrow(faithful)+nNoise, replace = TRUE, prob = c(3,1)/4)
mod5 <- Mclust(data, initialization = list(noise = NoiseInit))
summary(mod5, parameter = TRUE)
```
plot(mod5, what = "classification")

mclust.deprecated  Deprecated Functions in mclust package

Description

These functions are provided for compatibility with older versions of the mclust package only, and may be removed eventually.

Usage

cv.MclustDA(...)  
cv1EMtrain(data, labels, modelNames=NULL)  
bicEMtrain(data, labels, modelNames=NULL)

Arguments

...  pass arguments down.  
data  A numeric vector or matrix of observations.  
labels  Labels for each element or row in the dataset.  
modelNames  Vector of model names that should be tested. The default is to select all available model names.

See Also

deprecated

mclust.options  Default values for use with MCLUST package

Description

Set or retrieve default values for use with MCLUST package.

Usage

mclust.options(...)

Arguments

...  one or more arguments provided in the name = value form, or no argument at all may be given.  
Available arguments are described in the Details section below.
mclust.options

Details

mclust.options is provided for assigning or retrieving default values used by various functions in MCLUST.

Available options are:

emModelNames A vector of 3-character strings that are associated with multivariate models for which EM estimation is available in MCLUST.

The current default is all of the multivariate mixture models supported in MCLUST. The help file for mclustModelNames describes the available models.

hcModelName A string associated with multivariate models for which model-based hierarchical clustering is available in MCLUST.

The available models are the following:

"EII" spherical, equal volume
"EEE" ellipsoidal, equal volume, shape, and orientation
"VII" spherical, unequal volume
"VVV" ellipsoidal, varying volume, shape, and orientation.

The "VVV" is used as default for initialization of EM algorithm.

hcUse A string or a vector of character strings specifying the type of input variables to be used in model-based hierarchical clustering to start the EM algorithm. Possible values are:

"VARS" original variables;
"STD" standardized variables;
"SPH" sphered variables (centered, scaled, uncorrelated) computed using SVD;
"PCS" principal components computed using SVD on centered variables (i.e. using the covariance matrix);
"PCR" principal components computed using SVD on standardized (center and scaled) variables (i.e. using the correlation matrix);
"SVD" scaled SVD transformation;
"RND" no transformation is applied but a random hierarchical structure is returned (see hcRandomPairs).

For further details see Scrucca and Raftery (2015), Scrucca et al. (2016).

subset A value specifying the maximal sample size to be used in the model-based hierarchical clustering to start the EM algorithm. If data sample size exceeds this value, a random sample is drawn of size specified by subset.

fillEllipses A logical value specifying whether or not to fill with transparent colors ellipses corresponding to the within-cluster covariances in case of "classification" plot for 'Mclust' objects, or "scatterplot" graphs for 'MclustDA' objects.

bicPlotSymbols A vector whose entries correspond to graphics symbols for plotting the BIC values output from Mclust and mclustBIC. These are displayed in the legend which appears at the lower right of the BIC plots.

bicPlotColors A vector whose entries correspond to colors for plotting the BIC curves from output from Mclust and mclustBIC. These are displayed in the legend which appears at the lower right of the BIC plots.
classPlotSymbols A vector whose entries are either integers corresponding to graphics symbols or single characters for indicating classifications when plotting data. Classes are assigned symbols in the given order.

classPlotColors A vector whose entries correspond to colors for indicating classifications when plotting data. Classes are assigned colors in the given order.

warn A logical value indicating whether or not to issue certain warnings. Most of these warnings have to do with situations in which singularities are encountered. The default is warn = FALSE.

The parameter values set via a call to this function will remain in effect for the rest of the session, affecting the subsequent behaviour of the functions for which the given parameters are relevant.

Value

If the argument list is empty the function returns the current list of values. If the argument list is not empty, the returned list is invisible.

References


See Also

`Mclust, MclustDA, densityMclust, emControl`

Examples

```r
opt <- mclust.options() # save default values
irisBIC <- mclustBIC(iris[, -5])
summary(irisBIC, iris[, -5])
mclust.options(emModelNames = c("EII", "EEI", "EEE"))
irisBIC <- mclustBIC(iris[, -5])
summary(irisBIC, iris[, -5])
mclust.options(opt) # restore default values
mclust.options()

oldpar <- par(mfrow = c(2,1), no.readonly = TRUE)
n <- with(mclust.options(),
  max(sapply(list(bicPlotSymbols, bicPlotColors), length)))
plot(seq(n), rep(1, n), ylab = "", xlab = "", yaxt = "n",
  pch = mclust.options("bicPlotSymbols"),
  col = mclust.options("bicPlotColors"))
title("mclust.options("bicPlotSymbols") \n mclust.options("bicPlotColors")")
n <- with(mclust.options(),
  max(sapply(list(classPlotSymbols, classPlotColors), length)))
plot(seq(n), rep(1, n), ylab = "", xlab = "", yaxt = "n",
  pch = mclust.options("classPlotSymbols"),
```
mclust1Dplot

Plot one-dimensional data modeled by an MVN mixture.

Description

Plot one-dimensional data given parameters of an MVN mixture model for the data.

Usage

mclust1Dplot(data, parameters = NULL, z = NULL,
classification = NULL, truth = NULL, uncertainty = NULL,
what = c("classification", "density", "error", "uncertainty"),
symbols = NULL, colors = NULL, ngrid = length(data),
xlab = NULL, ylab = NULL,
xlim = NULL, ylim = NULL,
cex = 1, main = FALSE, ...)

Arguments

data A numeric vector of observations. Categorical variables are not allowed.
parameters A named list giving the parameters of an MCLUST model, used to produce superimposing ellipses on the plot. The relevant components are as follows:
pro Mixing proportions for the components of the mixture. There should one more mixing proportion than the number of Gaussian components if the mixture model includes a Poisson noise term.
mean The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.
variance A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.
z A matrix in which the [i,k]th entry gives the probability of observation i belonging to the kth class. Used to compute classification and uncertainty if those arguments aren't available.
classification A numeric or character vector representing a classification of observations (rows) of data. If present argument z will be ignored.
truth A numeric or character vector giving a known classification of each data point. If classification or z is also present, this is used for displaying classification errors.
uncertainty A numeric vector of values in (0,1) giving the uncertainty of each data point. If present argument z will be ignored.
Choose from one of the following options: "classification" (default), "density", "error", "uncertainty".

Either an integer or character vector assigning a plotting symbol to each unique class classification. Elements in symbols correspond to classes in classification in order of appearance in the observations (the order used by the function unique). The default is to use a single plotting symbol |. Classes are delineated by showing them in separate lines above the whole of the data.

Either an integer or character vector assigning a color to each unique class classification. Elements in colors correspond to classes in order of appearance in the observations (the order used by the function unique). The default is given is mclust.options("classPlotColors").

Number of grid points to use for density computation over the interval spanned by the data. The default is the length of the data set.

An argument specifying a label for the axes.

An argument specifying bounds of the plot. This may be useful for when comparing plots.

An argument specifying the size of the plotting symbols. The default value is 1.

A logical variable or NULL indicating whether or not to add a title to the plot identifying the dimensions used.

Other graphics parameters.

A plot showing location of the mixture components, classification, uncertainty, density and/or classification errors. Points in the different classes are shown in separated levels above the whole of the data.

mclust2Dplot, clPairs, coordProj

## Not run:
n <- 250  ## create artificial data
set.seed(1)
y <- c(rnorm(n,-5), rnorm(n,0), rnorm(n,5))
yclass <- c(rep(1,n), rep(2,n), rep(3,n))
yModel <- Mclust(y)
mclust1Dplot(y, parameters = yModel$parameters, z = yModel$z,
what = "classification")
mclust1Dplot(y, parameters = yModel$parameters, z = yModel$z,
what = "error", truth = yclass)
mclust1Dplot(y, parameters = yModel$parameters, z = yModel$z,"
mclust2Dplot

Plot two-dimensional data modelled by an MVN mixture

Description

Plot two-dimensional data given parameters of an MVN mixture model for the data.

Usage

mclust2Dplot(data, parameters = NULL, z = NULL, classification = NULL, truth = NULL, uncertainty = NULL, what = c("classification", "uncertainty", "error"), addEllipses = TRUE, fillEllipses = mclust.options("fillEllipses"), symbols = NULL, colors = NULL, xlim = NULL, ylim = NULL, xlab = NULL, ylab = NULL, scale = FALSE, cex = 1, PCH = ".", main = FALSE, swapAxes = FALSE, ...)

Arguments

data A numeric matrix or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables. In this case the data are two dimensional, so there are two columns.

parameters A named list giving the parameters of an MCLUST model, used to produce superimposing ellipses on the plot. The relevant components are as follows:

pro Mixing proportions for the components of the mixture. There should be one more mixing proportion than the number of Gaussian components if the mixture model includes a Poisson noise term.

mean The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.

variance A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.

z A matrix in which the [i,k]th entry gives the probability of observation i belonging to the kth class. Used to compute classification and uncertainty if those arguments aren’t available.
classification A numeric or character vector representing a classification of observations (rows) of data. If present argument z will be ignored.

truth A numeric or character vector giving a known classification of each data point. If classification or z is also present, this is used for displaying classification errors.

uncertainty A numeric vector of values in (0,1) giving the uncertainty of each data point. If present argument z will be ignored.

what Choose from one of the following three options: "classification" (default), "error", "uncertainty".

addEllipses A logical indicating whether or not to add ellipses with axes corresponding to the within-cluster covariances.

fillEllipses A logical specifying whether or not to fill ellipses with transparent colors when addEllipses = TRUE.

symbols Either an integer or character vector assigning a plotting symbol to each unique class in classification. Elements in colors correspond to classes in order of appearance in the sequence of observations (the order used by the function unique). The default is given by mclust.options("classPlotSymbols").

colors Either an integer or character vector assigning a color to each unique class in classification. Elements in colors correspond to classes in order of appearance in the sequence of observations (the order used by the function unique). The default is given is mclust.options("classPlotColors").

xlim, ylim Optional argument specifying bounds for the ordinate, abscissa of the plot. This may be useful for when comparing plots.

xlab, ylab Optional argument specifying labels for the x-axis and y-axis.

scale A logical variable indicating whether or not the two chosen dimensions should be plotted on the same scale, and thus preserve the shape of the distribution. Default: scale=FALSE

cex An argument specifying the size of the plotting symbols. The default value is 1.

PCH An argument specifying the symbol to be used when a classification has not been specified for the data. The default value is a small dot ".".

main A logical variable or NULL indicating whether or not to add a title to the plot identifying the dimensions used.

swapAxes A logical variable indicating whether or not the axes should be swapped for the plot.

... Other graphics parameters.

Value
A plot showing the data, together with the location of the mixture components, classification, uncertainty, and/or classification errors.

See Also
surfacePlot, clPairs, coordProj, mclust.options
mclustBIC

**BIC for Model-Based Clustering**

### Description

BIC for parameterized Gaussian mixture models fitted by EM algorithm initialized by model-based hierarchical clustering.

### Usage

```r
mclustBIC(data, G = NULL, modelNames = NULL,
          prior = NULL, control = emControl(),
          initialization = list(hcPairs = NULL,
                               subset = NULL,
                               noise = NULL),
          Vinv = NULL, warn = mclust.options("warn"),
          x = NULL, verbose = interactive(), ...)
```

### Arguments

- **data**
  A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

- **G**
  An integer vector specifying the numbers of mixture components (clusters) for which the BIC is to be calculated. The default is G=1:9, unless the argument `x` is specified, in which case the default is taken from the values associated with `x`.

- **modelNames**
  A vector of character strings indicating the models to be fitted in the EM phase of clustering. The help file for `mclustModelNames` describes the available models. The default is:
  
  c(“E”, “V”) for univariate data
mclust.options(“emModelNames”) for multivariate data (n > d)
c(“EII”, “VII”, “EEI”, “EVI”, “VEI”, “VVI”) the spherical and diagonal models for multivariate data (n <= d)
unless the argument \texttt{x} is specified, in which case the default is taken from the values associated with \texttt{x}.

\textbf{prior} \hspace{1cm} The default assumes no prior, but this argument allows specification of a conjugate prior on the means and variances through the function \texttt{priorControl}.

\textbf{control} \hspace{1cm} A list of control parameters for EM. The defaults are set by the call \texttt{emControl()}.  

\textbf{initialization} \hspace{1cm} A list containing zero or more of the following components:

- \texttt{hcPairs} A matrix of merge pairs for hierarchical clustering such as produced by function \texttt{hc}.
  For multivariate data, the default is to compute a hierarchical agglomerative clustering tree by applying function \texttt{hc} with model specified by \texttt{mclust.options("hcModelName")}, and data transformation set by \texttt{mclust.options("hcUse")}.
  All the input or a subset as indicated by the \texttt{subset} argument is used for initial clustering.
  The hierarchical clustering results are then used to start the EM algorithm from a given partition.
  For univariate data, the default is to use quantiles to start the EM algorithm. However, hierarchical clustering could also be used by calling \texttt{hc} with model specified as "V" or "E".

- \texttt{subset} A logical or numeric vector specifying a subset of the data to be used in the initial hierarchical clustering phase. By default no subset is used unless the number of observations exceeds the value specified by \texttt{mclust.options("subset")}.
  The \texttt{subset} argument is ignored if \texttt{hcPairs} are provided. Note that to guarantee exact reproducibility of results a seed must be specified (see \texttt{set.seed}).

- \texttt{noise} A logical or numeric vector indicating an initial guess as to which observations are noise in the data. If numeric the entries should correspond to row indexes of the data. If supplied, a noise term will be added to the model in the estimation.

- \texttt{Vinv} An estimate of the reciprocal hypervolume of the data region. The default is determined by applying function \texttt{hypvol} to the data. Used only if an initial guess as to which observations are noise is supplied.

- \texttt{warn} A logical value indicating whether or not certain warnings (usually related to singularity) should be issued when estimation fails. The default is controlled by \texttt{mclust.options}.

- \texttt{x} An object of class \texttt{’mclustBIC’}. If supplied, \texttt{mclustBIC} will use the settings in \texttt{x} to produce another object of class \texttt{’mclustBIC’}, but with \texttt{G} and \texttt{modelNames} as specified in the arguments. Models that have already been computed in \texttt{x} are not recomputed. All arguments to \texttt{mclustBIC} except \texttt{data}, \texttt{G} and \texttt{modelName} are ignored and their values are set as specified in the attributes of \texttt{x}. Defaults for \texttt{G} and \texttt{modelNames} are taken from \texttt{x}.

- \texttt{verbose} A logical controlling if a text progress bar is displayed during the fitting procedure. By default is \texttt{TRUE} if the session is interactive, and \texttt{FALSE} otherwise.

\texttt{...} Catches unused arguments in indirect or list calls via \texttt{do.call}. 

mclustBIC

Value

Return an object of class `mclustBIC` containing the Bayesian Information Criterion for the specified mixture models numbers of clusters. Auxiliary information returned as attributes.

The corresponding `print` method shows the matrix of values and the top models according to the BIC criterion.

References


See Also

priorControl, emControl, mclustModel, summary.mclustBIC, hc, me, mclustModelNames, mclust.options

Examples

```r
irisBIC <- mclustBIC(iris[, -5])
irisBIC
plot(irisBIC)

## Not run:
subset <- sample(1:nrow(iris), 100)
irisBIC <- mclustBIC(iris[, -5], initialization=list(subset = subset))
irisBIC
plot(irisBIC)

irisBIC1 <- mclustBIC(iris[, -5], G=seq(from=1, to=9, by=2),
                       modelNames=c("EII", "EEI", "EEE"))
irisBIC1
plot(irisBIC1)
irisBIC2 <- mclustBIC(iris[, -5], G=seq(from=2, to=8, by=2),
                       modelNames=c("VII", "VVI", "VVV"), x= irisBIC1)
irisBIC2
plot(irisBIC2)

## End(Not run)

nNoise <- 450
set.seed(0)
poissonNoise <- apply(apply(iris[, -5], 2, range), 2, function(x, n)
                      runif(n, min = x[1]-.1, max = x[2]+.1), n = nNoise)
set.seed(0)
oiseInit <- sample(c(TRUE, FALSE), size=nrow(iris)+nNoise, replace=TRUE,
                    prob=c(3, 1))
```
irisNdata <- rbind(iris[, -5], poissonNoise)
irisNbic <- mclustBIC(data = irisNdata, G = 1:5,
                       initialization = list(noise = noiseInit))
irisNbic
plot(irisNbic)

mclustBICupdate Update BIC values for parameterized Gaussian mixture models

Description
Update the BIC (Bayesian Information Criterion) for parameterized Gaussian mixture models by taking the best from BIC results as returned by `mclustBIC`.

Usage
mclustBICupdate(BIC, ...)

Arguments
BIC Object of class 'mclustBIC' containing the BIC values as returned by a call to `mclustBIC`.
...
Further objects of class 'mclustBIC' to be merged.

Value
An object of class 'mclustBIC' containing the best values obtained from merging the input arguments. Attributes are also updated according to the best BIC found, so calling `Mclust` on the resulting output will return the corresponding best model (see example).

See Also
`mclustBIC`, `Mclust`.

Examples
```r
## Not run:
data(galaxies, package = "MASS")
galaxies <- galaxies / 1000

# use several random starting points
BIC <- NULL
for(j in 1:100)
{
  rBIC <- mclustBIC(galaxies, verbose = FALSE,
                    initialization = list(hcPairs = hcRandomPairs(galaxies)))
  BIC <- mclustBICupdate(BIC, rBIC)
}
pickBIC(BIC)
```
plot(BIC)

mod <- Mclust(galaxies, x = BIC)
summary(mod)

## End(Not run)

### Description

Bootstrap or jackknife estimation of standard errors and percentile bootstrap confidence intervals for the parameters of a Gaussian mixture model.

### Usage

\[
\text{MclustBootstrap}(\text{object, nboot = 999, type = c("bs", "wlbs", "pb", "jk"), max.nonfit = 10*nboot, verbose = interactive(), ...})
\]

### Arguments

- **object**: An object of class 'Mclust' or 'densityMclust' providing an estimated Gaussian mixture model.
- **nboot**: The number of bootstrap replications.
- **type**: A character string specifying the type of resampling to use:
  - "bs" nonparametric bootstrap
  - "wlbs" weighted likelihood bootstrap
  - "pb" parametric bootstrap
  - "jk" jackknife
- **max.nonfit**: The maximum number of non-estimable models allowed.
- **verbose**: A logical controlling if a text progress bar is displayed during the resampling procedure. By default is TRUE if the session is interactive, and FALSE otherwise.
- ... Further arguments passed to or from other methods.

### Details

For a fitted Gaussian mixture model with \text{object}$^{G}$ mixture components and covariances parameterisation \text{object}$^{G}$\text{modelName}, this function returns either the bootstrap distribution or the jackknife distribution of mixture parameters. In the former case, the nonparametric bootstrap or the weighted likelihood bootstrap approach could be used, so the the bootstrap procedure generates \text{nboot} bootstrap samples of the same size as the original data by resampling with replacement from the observed data. In the jackknife case, the procedure considers all the samples obtained by omitting one observation at time.

The resulting resampling distribution can then be used to obtain standard errors and percentile confidence intervals by the use of \text{summary.MclustBootstrap} function.
Value
An object of class 'MclustBootstrap' with the following components:

- **n**
The number of observations in the data.

- **d**
The dimension of the data.

- **G**
A value specifying the number of mixture components.

- **modelName**
A character string specifying the mixture model covariances parameterisation (see mclustModelNames).

- **parameters**
A list of estimated parameters for the mixture components with the following components:
  - **pro**
a vector of mixing proportions.
  - **mean**
a matrix of means for each component.
  - **variance**
an array of covariance matrices for each component.

- **nboot**
The number of bootstrap replications if type = "bs" or type = "wlbs". The sample size if type = "jk".

- **type**
The type of resampling approach used.

- **nonfit**
The number of resamples that did not convergence during the procedure.

- **pro**
A matrix of dimension (nboot x G) containing the bootstrap distribution for the mixing proportion.

- **mean**
An array of dimension (nboot x d x G), where d is the dimension of the data, containing the bootstrap distribution for the component means.

- **variance**
An array of dimension (nboot x d x d x G), where d is the dimension of the data, containing the bootstrap distribution for the component covariances.

References


See Also


Examples

```r
## Not run:
data(diabetes)
X <- diabetes[, -1]
modClust <- Mclust(X)
bootClust <- MclustBootstrap(modClust)
summary(bootClust, what = "se")
summary(bootClust, what = "ci")
```
mclustBootstrapLRT

Bootstrap Likelihood Ratio Test for the Number of Mixture Components

Description

Perform the likelihood ratio test (LRT) for assessing the number of mixture components in a specific finite mixture model parameterisation. The observed significance is approximated by using the (parametric) bootstrap for the likelihood ratio test statistic (LRTS).

Usage

mclustBootstrapLRT(data, modelName = NULL, nboot = 999, level = 0.05, maxG = NULL, verbose = interactive(), ...)

## S3 method for class `mclustBootstrapLRT`
print(x, ...)

## S3 method for class `mclustBootstrapLRT`
plot(x, G = 1, hist.col = "grey", hist.border = "lightgrey", breaks = "Scott", col = "forestgreen", lwd = 2, lty = 3, main = NULL, ...)

Arguments

data A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

modelName A character string indicating the mixture model to be fitted. The help file for mclustModelNames describes the available models.

nboot The number of bootstrap replications to use (by default 999).

level The significance level to be used to terminate the sequential bootstrap procedure.

maxG The maximum number of mixture components $G$ to test. If not provided the procedure is stopped when a test is not significant at the specified level.

verbose A logical controlling if a text progress bar is displayed during the bootstrap procedure. By default is TRUE if the session is interactive, and FALSE otherwise.

... Further arguments passed to or from other methods. In particular, see the optional arguments in mclustBIC.
mclustBootstrapLRT

x

An 'mclustBootstrapLRT' object.

G

A value specifying the number of components for which to plot the bootstrap distribution.

hist.col

The colour to be used to fill the bars of the histogram.

hist.border

The color of the border around the bars of the histogram.

breaks

See the argument in function hist.

col, lwd, lty

The color, line width and line type to be used to represent the observed LRT statistic.

main

The title for the graph.

Details

The implemented algorithm for computing the LRT observed significance using the bootstrap is the following. Let $G_0$ be the number of mixture components under the null hypothesis versus $G_1 = G_0 + 1$ under the alternative. Bootstrap samples are drawn by simulating data under the null hypothesis. Then, the p-value may be approximated using eq. (13) on McLachlan and Rathnayake (2014). Equivalently, using the notation of Davison and Hinkley (1997) it may be computed as

$$p\text{-value} = \frac{1 + \#\{LRT_{b}^{*} \geq LRT_{obs}\}}{B + 1}$$

where

$B =$ number of bootstrap samples

$LRT_{obs}$ = LRTS computed on the observed data

$LRT_{b}^{*}$ = LRTS computed on the $b$th bootstrap sample.

Value

An object of class 'mclustBootstrapLRT' with the following components:

G

A vector of number of components tested under the null hypothesis.

modelName

A character string specifying the mixture model as provided in the function call (see above).

obs

The observed values of the LRTS.

boot

A matrix of dimension nboot x the number of components tested containing the bootstrap values of LRTS.

p.value

A vector of p-values.

References


MclustDA

MclustDA discriminant analysis

Description

Discriminant analysis based on Gaussian finite mixture modeling.

Usage

MclustDA(data, class, G = NULL, modelNames = NULL,
modelType = c("MclustDA", "EDDA"),
prior = NULL,
control = emControl(),
initialization = NULL,
warn = mclust.options("warn"),
verbose = interactive(),
...)  

Arguments

data
A data frame or matrix giving the training data.

class
A vector giving the known class labels (either a numerical value or a character string) for the observations in the training data.

G
An integer vector specifying the numbers of mixture components (clusters) for which the BIC is to be calculated within each class. The default is G = 1:5.  
A different set of mixture components for each class can be specified by providing this argument with a list of integers for each class. See the examples below.

modelNames
A vector of character strings indicating the models to be fitted by EM within each class (see the description in mclustModelNames). A different set of mixture models for each class can be specified by providing this argument with a list of character strings. See the examples below.

See Also

mclustBIC, mclustICL, Mclust

Examples

## Not run:
data(faithful)
faithful.boot = mclustBootstrapLRT(faithful, model = "VVV")
faithful.boot
plot(faithful.boot, G = 1)
plot(faithful.boot, G = 2)
## End(Not run)
modelType A character string specifying whether the models given in modelNames should fit a different number of mixture components and covariance structures for each class ("MclustDA", the default) or should be constrained to have a single component for each class with the same covariance structure among classes ("EDDA"). See Details section and the examples below.

prior The default assumes no prior, but this argument allows specification of a conjugate prior on the means and variances through the function priorControl.

control A list of control parameters for EM. The defaults are set by the call emControl().

initialization A list containing zero or more of the following components:

hcPairs A matrix of merge pairs for hierarchical clustering such as produced by function hc. The default is to compute a hierarchical clustering tree by applying function hc with modelName = "E" to univariate data and modelName = "VVV" to multivariate data or a subset as indicated by the subset argument. The hierarchical clustering results are used as starting values for EM.

subset A logical or numeric vector specifying a subset of the data to be used in the initial hierarchical clustering phase.

warn A logical value indicating whether or not certain warnings (usually related to singularity) should be issued when estimation fails. The default is controlled by mclust.options.

verbose A logical controlling if a text progress bar is displayed during the fitting procedure. By default is TRUE if the session is interactive, and FALSE otherwise.

... Further arguments passed to or from other methods.

Details

The "EDDA" method for discriminant analysis is described in Bensmail and Celeux (1996), while "MclustDA" in Fraley and Raftery (2002).

Value

An object of class 'MclustDA' providing the optimal (according to BIC) mixture model.

The details of the output components are as follows:

call The matched call.
data The input data matrix.
class The input class labels.
type A character string specifying the modelType estimated.
models A list of Mclust objects containing information on fitted model for each class.
n The total number of observations in the data.
d The dimension of the data.
bic Optimal BIC value.
loglik Log-likelihood for the selected model.
df Number of estimated parameters.
MclustDA

Author(s)
Luca Scrucca

References

See Also

Examples
odd <- seq(from = 1, to = nrow(iris), by = 2)
even <- odd + 1
X.train <- iris[odd,-5]
Class.train <- iris[odd,5]
X.test <- iris[even,-5]
Class.test <- iris[even,5]

# common EEE covariance structure (which is essentially equivalent to linear discriminant analysis)
irisMclustDA <- MclustDA(X.train, Class.train, modelType = "EDDA", modelNames = "EEE")
summary(irisMclustDA, parameters = TRUE)
summary(irisMclustDA, newdata = X.test, newclass = Class.test)

# common covariance structure selected by BIC
irisMclustDA <- MclustDA(X.train, Class.train, modelType = "EDDA")
summary(irisMclustDA, parameters = TRUE)
summary(irisMclustDA, newdata = X.test, newclass = Class.test)

# general covariance structure selected by BIC
irisMclustDA <- MclustDA(X.train, Class.train)
summary(irisMclustDA, parameters = TRUE)
summary(irisMclustDA, newdata = X.test, newclass = Class.test)

plot(irisMclustDA)
plot(irisMclustDA, dimens = 3:4)
plot(irisMclustDA, dimens = 4)

plot(irisMclustDA, what = "classification")
plot(irisMclustDA, what = "classification", newdata = X.test)
plot(irisMclustDA, what = "classification", dimens = 3:4)
plot(irisMclustDA, what = "classification", newdata = X.test, dimens = 3:4)
plot(irisMclustDA, what = "classification", dimens = 4)
plot(irisMclustDA, what = "classification", dimens = 4, newdata = X.test)
plot(irisMclustDA, what = "train&test", newdata = X.test)
plot(irisMclustDA, what = "train&test", newdata = X.test, dimens = 3:4)
plot(irisMclustDA, what = "train&test", newdata = X.test, dimens = 4)
plot(irisMclustDA, what = "error")
plot(irisMclustDA, what = "error", dimens = 3:4)
plot(irisMclustDA, what = "error", dimens = 4)
plot(irisMclustDA, what = "error", newdata = X.test, newclass = Class.test)
plot(irisMclustDA, what = "error", newdata = X.test, newclass = Class.test, dimens = 3:4)
plot(irisMclustDA, what = "error", newdata = X.test, newclass = Class.test, dimens = 4)

## Not run:
# simulated 1D data
n <- 250
set.seed(1)
triModal <- c(rnorm(n,-5), rnorm(n,0), rnorm(n,5))
triClass <- c(rep(1,n), rep(2,n), rep(3,n))
odd <- seq(from = 1, to = length(triModal), by = 2)
even <- odd + 1
triMclustDA <- MclustDA(triModal[odd], triClass[odd])
summary(triMclustDA, parameters = TRUE)
summary(triMclustDA, newdata = triModal[even], newclass = triClass[even])
plot(triMclustDA, what = "scatterplot")
plot(triMclustDA, what = "classification")
plot(triMclustDA, what = "classification", newdata = triModal[even])
plot(triMclustDA, what = "train&test", newdata = triModal[even])
plot(triMclustDA, what = "error")
plot(triMclustDA, what = "error", newdata = triModal[even], newclass = triClass[even])

# simulated 2D cross data
data(cross)
odd <- seq(from = 1, to = nrow(cross), by = 2)
even <- odd + 1
crossMclustDA <- MclustDA(cross[odd,-1], cross[odd,1])
summary(crossMclustDA, parameters = TRUE)
summary(crossMclustDA, newdata = cross[even,-1], newclass = cross[even,1])
plot(crossMclustDA, what = "scatterplot")
plot(crossMclustDA, what = "classification")
plot(crossMclustDA, what = "classification", newdata = cross[even,-1])
plot(crossMclustDA, what = "train&test", newdata = cross[even,-1])
plot(crossMclustDA, what = "error")
plot(crossMclustDA, what = "error", newdata = cross[even,-1], newclass = cross[even,1])

## End(Not run)
Description
A dimension reduction method for visualizing the clustering or classification structure obtained from a finite mixture of Gaussian densities.

Usage
```r
MclustDR(object, lambda = 0.5, normalized = TRUE, Sigma, tol = sqrt(.Machine$double.eps))
```

Arguments
- `object`: An object of class 'Mclust' or 'MclustDA' resulting from a call to, respectively, `Mclust` or `MclustDA`.
- `lambda`: A tuning parameter in the range [0,1] described in Scrucca (2014). The default 0.5 gives equal importance to differences in means and covariances among clusters/classes. To recover the directions that mostly separate the estimated clusters or classes set this parameter to 1.
- `normalized`: Logical. If TRUE directions are normalized to unit norm.
- `Sigma`: Marginal covariance matrix of data. If not provided is estimated by the MLE of observed data.
- `tol`: A tolerance value.

Details
The method aims at reducing the dimensionality by identifying a set of linear combinations, ordered by importance as quantified by the associated eigenvalues, of the original features which capture most of the clustering or classification structure contained in the data.

Information on the dimension reduction subspace is obtained from the variation on group means and, depending on the estimated mixture model, on the variation on group covariances (see Scrucca, 2010).

Observations may then be projected onto such a reduced subspace, thus providing summary plots which help to visualize the underlying structure.

The method has been extended to the supervised case, i.e. when the true classification is known (see Scrucca, 2013).

This implementation doesn’t provide a formal procedure for the selection of dimensionality. A future release will include one or more methods.

Value
An object of class 'MclustDR' with the following components:
- `call`: The matched call
- `type`: A character string specifying the type of model for which the dimension reduction is computed. Currently, possible values are "Mclust" for clustering, and "MclustDA" or "EDDA" for classification.
- `x`: The data matrix.
Sigma  The covariance matrix of the data.

mixcomp  A numeric vector specifying the mixture component of each data observation.

class  A factor specifying the classification of each data observation. For model-based clustering this is equivalent to the corresponding mixture component. For model-based classification this is the known classification.

G  The number of mixture components.

modelName  The name of the parameterization of the estimated mixture model(s). See mclustModelNames.

mu  A matrix of means for each mixture component.

sigma  An array of covariance matrices for each mixture component.

pro  The estimated prior for each mixture component.

M  The kernel matrix.

lambda  The tuning parameter.

evalues  The eigenvalues from the generalized eigen-decomposition of the kernel matrix.

raw.evectors  The raw eigenvectors from the generalized eigen-decomposition of the kernel matrix, ordered according to the eigenvalues.

basis  The basis of the estimated dimension reduction subspace.

std.basis  The basis of the estimated dimension reduction subspace standardized to variables having unit standard deviation.

numdir  The dimension of the projection subspace.

dir  The estimated directions, i.e. the data projected onto the estimated dimension reduction subspace.

Author(s)

Luca Scrucca

References


See Also

summary.MclustDR, plot.MclustDR, Mclust, MclustDA.

Examples

# clustering
data(diabetes)
mod <- Mclust(diabetes[, -1])
summary(mod)

dr <- MclustDR(mod)
```r
summary(dr)
plot(dr, what = "scatterplot")
plot(dr, what = "evalues")

# adjust the tuning parameter to show the most separating directions
dr1 <- MclustDR(mod, lambda = 1)
summary(dr1)
plot(dr1, what = "scatterplot")
plot(dr1, what = "evalues")

# classification
data(banknote)
da <- MclustDA(banknote[,2:7], banknote$Status, modelType = "EDDA")
dr <- MclustDR(da)
summary(dr)
da <- MclustDA(banknote[,2:7], banknote$Status)
dr <- MclustDR(da)
summary(dr)
```

---

### MclustDRsubsel

**Subset selection for GMMDR directions based on BIC**

#### Description

Implements a subset selection method for selecting the relevant directions spanning the dimension reduction subspace for visualizing the clustering or classification structure obtained from a finite mixture of Gaussian densities.

#### Usage

```r
MclustDRsubsel(object, G = 1:9,
modelNames = mclust.options("emModelNames"),
...,  
bic.stop = 0, bic.cutoff = 0,
mindir = 1,
verbose = interactive())
```

#### Arguments

- **object** An object of class 'MclustDR' resulting from a call to `MclustDR`.
- **G** An integer vector specifying the numbers of mixture components or clusters.
- **modelNames** A vector of character strings indicating the models to be fitted. See `mclustModelNames` for a description of the available models.
- **...** Further arguments passed through `Mclust` or `MclustDA`.
bic.stop A criterion to terminate the search. If maximal BIC difference is less than bic.stop then the algorithm stops.
Two typical values are:
0: algorithm stops when the BIC difference becomes negative (default)
-Inf: algorithm continues until all directions have been selected

bic.cutoff A value specifying how to select simplest “best” model within bic.cutoff from the maximum value achieved. Setting this to 0 (default) simply select the model with the largest BIC difference.

mindir An integer value specifying the minimum number of directions to be estimated.

verbose A logical or integer value specifying if and how much detailed information should be reported during the iterations of the algorithm.
Possible values are:
0 or FALSE: no trace info is shown;
1 or TRUE: a trace info is shown at each step of the search;
2: a more detailed trace info is is shown.

Details
The GMMDR method aims at reducing the dimensionality by identifying a set of linear combinations, ordered by importance as quantified by the associated eigenvalues, of the original features which capture most of the clustering or classification structure contained in the data. This is implemented in MclustDRsubsel.

The MclustDRsubsel function implements the greedy forward search algorithm discussed in Scrucca (2010) to prune the set of all GMMDR directions. The criterion used to select the relevant directions is based on the BIC difference between a clustering model and a model in which the feature proposal has no clustering relevance. The steps are the following:
1. Select the first feature to be the one which maximizes the BIC difference between the best clustering model and the model which assumes no clustering, i.e. a single component.
2. Select the next feature amongst those not previously included, to be the one which maximizes the BIC difference.
3. Iterate the previous step until all the BIC differences for the inclusion of a feature become less than bic.stop.
At each step, the search over the model space is performed with respect to the model parametrisation and the number of clusters.

Value
An object of class 'MclustDRsubsel' which inherits from 'MclustDR', so it has the same components of the latter plus the following:

basisx The basis of the estimated dimension reduction subspace expressed in terms of the original variables.

std.basisx The basis of the estimated dimension reduction subspace expressed in terms of the original variables standardized to have unit standard deviation.
Author(s)
Luca Scrucca

References

See Also
MclustDR, Mclust, MclustDA.

Examples
```r
## Not run:
# clustering
data(crabs, package = "MASS")
x <- crabs[,4:8]
class <- paste(crabs$sp, crabs$sex, sep = "|")
mod <- Mclust(x)
table(class, mod$classification)
dr <- MclustDR(mod)
summary(dr)
plot(dr)
drs <- MclustDRsubsel(dr)
summary(drs)
table(class, drs$classification)
plot(drs, what = "scatterplot")
plot(drs, what = "pairs")
plot(drs, what = "contour")
plot(drs, what = "boundaries")
plot(drs, what = "evalues")

# classification
data(banknote)
da <- MclustDA(banknote[,2:7], banknote$Status)
table(banknote$Status, predict(da)$class)
dr <- MclustDR(da)
summary(dr)
drs <- MclustDRsubsel(dr)
summary(drs)
table(banknote$Status, predict(drs)$class)
plot(drs, what = "scatterplot")
plot(drs, what = "classification")
plot(drs, what = "boundaries")
## End(Not run)
```
mclustICL

ICL Criterion for Model-Based Clustering

Description

ICL (Integrated Complete-data Likelihood) for parameterized Gaussian mixture models fitted by EM algorithm initialized by model-based hierarchical clustering.

Usage

mclustICL(data, G = NULL, modelNames = NULL,
          initialization = list(hcPairs = NULL,
                               subset = NULL,
                               noise = NULL),
          x = NULL, ...)

## S3 method for class 'mclustICL'
summary(object, G, modelNames, ...)

Arguments

data  A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

G  An integer vector specifying the numbers of mixture components (clusters) for which the criteria should be calculated. The default is G = 1:9.

modelNames  A vector of character strings indicating the models to be fitted in the EM phase of clustering. The help file for mclustModelNames describes the available models. The default is:
c("E", "V") for univariate data
mclust.options("emModelNames") for multivariate data (n > d)
c("EII", "VII", "EEI", "EVI", "VEI", "VVI") the spherical and diagonal models for multivariate data (n <= d)

initialization  A list containing zero or more of the following components:

hcPairs  A matrix of merge pairs for hierarchical clustering such as produced by function hc. For multivariate data, the default is to compute a hierarchical clustering tree by applying function hc with modelName = "VVV" to the data or a subset as indicated by the subset argument. The hierarchical clustering results are to start EM. For univariate data, the default is to use quantiles to start EM.

subset  A logical or numeric vector specifying a subset of the data to be used in the initial hierarchical clustering phase.

x  An object of class 'mclustICL'. If supplied, mclustICL will use the settings in x to produce another object of class 'mclustICL', but with G and modelNames as specified in the arguments. Models that have already been computed in x are
not recomputed. All arguments to \texttt{mclustICL} except \texttt{data}, \texttt{G} and \texttt{modelName}
are ignored and their values are set as specified in the attributes of \texttt{x}. Defaults
for \texttt{G} and \texttt{modelName} are taken from \texttt{x}.

\ldots

Further arguments used in the call to \texttt{Mclust}. See also \texttt{mclustBIC}.

\texttt{object}

An integer vector specifying the numbers of mixture components (clusters) for
which the criteria should be calculated. The default is \texttt{G = 1:9}.

\textbf{Value}

Returns an object of class 'mclustICL' containing the the ICL criterion for the specified mixture
models and numbers of clusters.

The corresponding \texttt{print} method shows the matrix of values and the top models according to the
ICL criterion. The \texttt{summary} method shows only the top models.

\textbf{References}

Biernacki, C., Celeux, G., Govaert, G. (2000). Assessing a mixture model for clustering with the
719-725.

Scrucca L., Fop M., Murphy T. B. and Raftery A. E. (2016) mclust 5: clustering, classification and

\textbf{See Also}

\texttt{plot.mclustICL,Mclust,mclustBIC,mclustBootstrapLRT,bic,icl}

\textbf{Examples}

\begin{verbatim}
data(faithful)
faithful.ICL <- mclustICL(faithful)
faithful.ICL
summary(faithful.ICL)
plot(faithful.ICL)
## Not run:
# compare with
faithful.BIC <- mclustBIC(faithful)
faithful.BIC
plot(faithful.BIC)
## End(Not run)
\end{verbatim}
mclustLoglik

Log-likelihood from a table of BIC values for parameterized Gaussian mixture models

Description

Compute the maximal log-likelihood from a table of BIC values contained in a 'mclustBIC' object as returned by function mclustBIC.

Usage

mclustLoglik(object, ...)

Arguments

object
An object of class 'mclustBIC' containing the BIC values as returned by a call to mclustBIC.

...
Catches unused arguments in an indirect or list call via do.call.

Value

An object of class 'mclustLoglik' containing the maximal log-likelihood values for the Gaussian mixture models provided as input.

See Also

mclustBIC.

Examples

## Not run:
BIC <- mclustBIC(iris[,1:4])
mclustLoglik(BIC)
## End(Not run)

mclustModel

Best model based on BIC

Description

Determines the best model from clustering via mclustBIC for a given set of model parameterizations and numbers of components.

Usage

mclustModel(data, BICvalues, G, modelName, ...)

Arguments

...
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>The matrix or vector of observations used to generate ‘object’.</td>
</tr>
<tr>
<td>BICvalues</td>
<td>An ‘mclustBIC’ object, which is the result of applying mclustBIC to data.</td>
</tr>
<tr>
<td>G</td>
<td>A vector of integers giving the numbers of mixture components (clusters) from which the best model according to BIC will be selected (as character(G) must be a subset of the row names of BICvalues). The default is to select the best model for all numbers of mixture components used to obtain BICvalues.</td>
</tr>
<tr>
<td>modelNames</td>
<td>A vector of integers giving the model parameterizations from which the best model according to BIC will be selected (as character(model) must be a subset of the column names of BICvalues). The default is to select the best model for parameterizations used to obtain BICvalues.</td>
</tr>
</tbody>
</table>

... Not used. For generic/method consistency.

Value

A list giving the optimal (according to BIC) parameters, conditional probabilities $z$, and log-likelihood, together with the associated classification and its uncertainty.

The details of the output components are as follows:

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>modelName</td>
<td>A character string indicating the model. The help file for mclustModelNames describes the available models.</td>
</tr>
<tr>
<td>n</td>
<td>The number of observations in the data.</td>
</tr>
<tr>
<td>d</td>
<td>The dimension of the data.</td>
</tr>
<tr>
<td>G</td>
<td>The number of components in the Gaussian mixture model corresponding to the optimal BIC.</td>
</tr>
<tr>
<td>bic</td>
<td>The optimal BIC value.</td>
</tr>
<tr>
<td>loglik</td>
<td>The log-likelihood corresponding to the optimal BIC.</td>
</tr>
<tr>
<td>parameters</td>
<td>A list with the following components:</td>
</tr>
<tr>
<td>pro</td>
<td>A vector whose $k$th component is the mixing proportion for the $k$th component of the mixture model. If missing, equal proportions are assumed.</td>
</tr>
<tr>
<td>mean</td>
<td>The mean for each component. If there is more than one component, this is a matrix whose $k$th column is the mean of the $k$th component of the mixture model.</td>
</tr>
<tr>
<td>variance</td>
<td>A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.</td>
</tr>
<tr>
<td>Vinv</td>
<td>The estimate of the reciprocal hypervolume of the data region used in the computation when the input indicates the addition of a noise component to the model.</td>
</tr>
<tr>
<td>z</td>
<td>A matrix whose $i,k$th entry is the probability that observation $i$ in the test data belongs to the $k$th class.</td>
</tr>
</tbody>
</table>

See Also

mclustBIC
Examples

irisBIC <- mclustBIC(iris[, -5])
mclustModel(iris[, -5], irisBIC)
mclustModel(iris[, -5], irisBIC, G = 1:6, modelNames = c("VII", "VVI", "VVV"))

mclustModelNames

Description

Description of model names used in the MCLUST package.

Usage

mclustModelNames(model)

Arguments

model A string specifying the model.

Details

The following models are available in package mclust:

univariate mixture

"E" equal variance (one-dimensional)
"V" variable/unequal variance (one-dimensional)

multivariate mixture

"EII" spherical, equal volume
"VII" spherical, unequal volume
"EEI" diagonal, equal volume and shape
"VEI" diagonal, varying volume, equal shape
"EVI" diagonal, equal volume, varying shape
"VVI" diagonal, varying volume and shape
"EEE" ellipsoidal, equal volume, shape, and orientation
"VEE" ellipsoidal, equal shape and orientation (*)
"EVE" ellipsoidal, equal volume and orientation (*)
"VVE" ellipsoidal, equal orientation (*)
"EEV" ellipsoidal, equal volume and equal shape
"VEV" ellipsoidal, equal shape
"EVV" ellipsoidal, equal volume (*)
"VVV" ellipsoidal, varying volume, shape, and orientation

Single component

"X" univariate normal
"XII" spherical multivariate normal
"XXI" diagonal multivariate normal
"XXX" ellipsoidal multivariate normal

(*) new models in mclust version >= 5.0.0.

Value

Returns a list with the following components:

- **model**: a character string indicating the model (as in input).
- **type**: the description of the indicated model (see Details section).

See Also

Mclust, mclustBIC

Examples

mclustModelNames("E")
mclustModelNames("EEE")
mclustModelNames("VVV")
mclustModelNames("XXI")
Arguments

- **data**: A data frame or matrix giving the training data.
- **class**: A vector giving the known class labels (either a numerical value or a character string) for the observations in the training data. Observations with unknown class are encoded as NA.
- **G**: An integer value specifying the numbers of mixture components or classes. By default is set equal to the number of known classes. See the examples below.
- **modelNames**: A vector of character strings indicating the models to be fitted by EM (see the description in `mclustModelNames`). See the examples below.
- **prior**: The default assumes no prior, but this argument allows specification of a conjugate prior on the means and variances through the function `priorControl`.
- **control**: A list of control parameters for EM. The defaults are set by the call `emControl()`.
- **warn**: A logical value indicating whether or not certain warnings (usually related to singularity) should be issued when estimation fails. The default is controlled by `mclust.options`.
- **verbose**: A logical controlling if a text progress bar is displayed during the fitting procedure. By default is TRUE if the session is interactive, and FALSE otherwise.
- **...**: Further arguments passed to or from other methods.

Details

The semi-supervised approach implemented in `MclustSSC()` is a simple Gaussian mixture model for classification where at the first M-step only observations with known class labels are used for parameters estimation. Then, a standard EM algorithm is used for updating the probability of class membership for unlabelled data while keeping fixed the probabilities for labelled data.

Value

An object of class 'MclustSSC' providing the optimal (according to BIC) Gaussian mixture model for semi-supervised classification.

The details of the output components are as follows:

- **call**: The matched call.
- **data**: The input data matrix.
- **class**: The input class labels (including NAs for unknown labels).
- **modelName**: A character string specifying the "best" estimated model.
- **G**: A numerical value specifying the number of mixture components or classes of the "best" estimated model.
- **n**: The total number of observations in the data.
- **d**: The dimension of the data.
- **BIC**: All BIC values.
- **loglik**: Log-likelihood for the selected model.
- **df**: Number of estimated parameters.
bic

Optimal BIC value.

parameters

A list with the following components:

pro  A vector whose $k$th component is the mixing proportion for the $k$th component of the mixture model.

mean  The mean for each component. If there is more than one component, this is a matrix whose $k$th column is the mean of the $k$th component of the mixture model.

variance  A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.

z  A matrix whose $[i,k]$th entry is the probability that observation $i$ in the test data belongs to the $k$th class.

classification  The classification corresponding to $z$, i.e. map(z).

prior  The prior used (if any).

control  A list of control parameters used in the EM algorithm.

Author(s)

Luca Scrucca

References


See Also

summary.MclustSSC, plot.MclustSSC, predict.MclustSSC

Examples

# Simulate two overlapping groups
n <- 200
pars <- list(pro = c(0.5, 0.5),
            mean = matrix(c(-1,1), nrow = 2, ncol = 2, byrow = TRUE),
            variance = mclustVariance("EII", d = 2, G = 2))
pars$variance$sigmasq <- 1
data <- sim("EII", parameters = pars, n = n, seed = 12)
class <- data[,1]
X <- data[, -1]
clPairs(X, class, symbols = c(1,2), main = "Full classified data")

# Randomly remove labels
cl <- class; cl[sample(1:n, size = 195)] <- NA
table(cl, useNA = "ifany")
clPairs(X, ifelse(is.na(cl), 0, class),
symbols = c(0,16,17), colors = c("grey", 4, 2),
main = "Partially classified data")
# Fit semi-supervised classification model
mod_SSC <- MclustSSC(X, cl)
summary(mod_SSC, parameters = TRUE)

pred_SSC <- predict(mod_SSC)
table(Predicted = pred_SSC$classification, Actual = class)

ngrid <- 50
xgrid <- seq(-3, 3, length.out = ngrid)
ygrid <- seq(-4, 4.5, length.out = ngrid)
xygrid <- expand.grid(xgrid, ygrid)
pred_SSC <- predict(mod_SSC, newdata = xygrid)
col <- mclust.options("classPlotColors")[class]
pch <- class
pch[!is.na(cl)] = ifelse(cl[!is.na(cl)] == 1, 19, 17)
plot(X, pch = pch, col = col)
contour(xgrid, ygrid, matrix(pred_SSC$z[,1], ngrid, ngrid),
add = TRUE, levels = 0.5, drawlabels = FALSE, lty = 2, lwd = 2)

---

mclustVariance

Template for variance specification for parameterized Gaussian mixture models

Description

Specification of variance parameters for the various types of Gaussian mixture models.

Usage

mclustVariance(modelName, d = NULL, G = 2)

Arguments

modelName A character string specifying the model.
d A integer specifying the dimension of the data.
G An integer specifying the number of components in the mixture model.

Details

The variance component in the parameters list from the output to e.g. `me` or `mstep` or input to e.g. `estep` may contain one or more of the following arguments, depending on the model:

modelName A character string indicating the model.
d The dimension of the data.
G The number of components in the mixture model.
sigmasq for the one-dimensional models ("E", "V") and spherical models ("EII", "VII"). This is either a vector whose 4th component is the variance for the 4th component in the mixture model ("V" and "VII"), or a scalar giving the common variance for all components in the mixture model ("E" and "EII").
For the equal variance models "EII", "EEI", and "EEE". A $d$ by $d$ matrix giving the common covariance for all components of the mixture model.

cholSigma For the equal variance model "EEE". A $d$ by $d$ upper triangular matrix giving the Cholesky factor of the common covariance for all components of the mixture model.

sigma For all multidimensional mixture models. A $d$ by $d$ by $G$ matrix array whose $[,,k]$th entry is the covariance matrix for the $k$th component of the mixture model.

cholSigma For the unconstrained covariance mixture model "VVV". A $d$ by $d$ by $G$ matrix array whose $[,,k]$th entry is the upper triangular Cholesky factor of the covariance matrix for the $k$th component of the mixture model.

scale For diagonal models "EEI", "EVI", "VEI", "VVI" and constant-shape models "EEV" and "VEV". Either a $G$-vector giving the scale of the covariance (the $d$th root of its determinant) for each component in the mixture model, or a single numeric value if the scale is the same for each component.

shape For diagonal models "EEI", "EVI", "VEI", "VVI" and constant-shape models "EEV" and "VEV". Either a $G$ by $d$ matrix in which the $k$th column is the shape of the covariance matrix (normalized to have determinant 1) for the $k$th component, or a $d$-vector giving a common shape for all components.

orientation For the constant-shape models "EEV" and "VEV". Either a $d$ by $d$ by $G$ array whose $[,,k]$th entry is the orthonormal matrix whose columns are the eigenvectors of the covariance matrix of the $k$th component, or a $d$ by $d$ orthonormal matrix if the mixture components have a common orientation. The orientation component is not needed in spherical and diagonal models, since the principal components are parallel to the coordinate axes so that the orientation matrix is the identity.

In all cases, the value $-1$ is used as a placeholder for unknown nonzero entries.

---

**me**  
*EM algorithm starting with M-step for parameterized MVN mixture models*

---

**Description**

Implements the EM algorithm for MVN mixture models parameterized by eigenevalue decomposition, starting with the maximization step.

**Usage**

```r
me(data, modelName, z, prior = NULL, control = emControl(),
   Vinv = NULL, warn = NULL, ...)
```
Arguments

data  A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

modelName  A character string indicating the model. The help file for \texttt{mclustModelNames} describes the available models.

z  A matrix whose $[i,k]$th entry is an initial estimate of the conditional probability of the $i$th observation belonging to the $k$th component of the mixture.

prior  Specification of a conjugate prior on the means and variances. See the help file for \texttt{priorControl} for further information. The default assumes no prior.

control  A list of control parameters for EM. The defaults are set by the call \texttt{emControl()}.

Vinv  If the model is to include a noise term, $V_{inv}$ is an estimate of the reciprocal hypervolume of the data region. If set to a negative value or 0, the model will include a noise term with the reciprocal hypervolume estimated by the function \texttt{hypvol}. The default is not to assume a noise term in the model through the setting $V_{inv}=\texttt{NULL}$.

warn  A logical value indicating whether or not certain warnings (usually related to singularity) should be issued when the estimation fails. The default is set in \texttt{mclust.options("warn")}.

...  Catches unused arguments in indirect or list calls via \texttt{do.call}.

Value

A list including the following components:

modelName  A character string identifying the model (same as the input argument).

n  The number of observations in the data.

d  The dimension of the data.

G  The number of mixture components.

z  A matrix whose $[i,k]$th entry is the conditional probability of the $i$th observation belonging to the $k$th component of the mixture.

parameters  A vector whose $k$th component is the mixing proportion for the $k$th component of the mixture model. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.

mean  The mean for each component. If there is more than one component, this is a matrix whose $k$th column is the mean of the $k$th component of the mixture model.

variance  A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for \texttt{mclustVariance} for details.

Vinv  The estimate of the reciprocal hypervolume of the data region used in the computation when the input indicates the addition of a noise component to the model.
The log likelihood for the data in the mixture model.

control
The list of control parameters for EM used.

prior
The specification of a conjugate prior on the means and variances used, NULL if no prior is used.

Attributes:  "info" Information on the iteration.
            "WARNING" An appropriate warning if problems are encountered in the computations.

See Also
   meE, ..., meVVV, em, mstep, estep, priorControl, mclustModelNames, mclustVariance, mclust.options

Examples
   ## Not run:
   me(modelName = "VVV", data = iris[, -5], z = unmap(iris[,5]))
   ## End(Not run)

me.weighted
EM algorithm with weights starting with M-step for parameterized MVN mixture models

Description
   Implements the EM algorithm for fitting MVN mixture models parameterized by eigenvalue
decomposition, when observations have weights, starting with the maximization step.

Usage
   me.weighted(modelName, data, z, weights = NULL, prior = NULL,
               control = emControl(), Vinv = NULL, warn = NULL, ...)

Arguments
   modelName
   A character string indicating the model. The help file for mclustModelNames
   describes the available models.

   data
   A numeric vector, matrix, or data frame of observations. Categorical variables
   are not allowed. If a matrix or data frame, rows correspond to observations and
   columns correspond to variables.

   z
   A matrix whose [i,k]th entry is an initial estimate of the conditional probability
   of the ith observation belonging to the kth component of the mixture.

   weights
   A vector of positive weights, where the [i]th entry is the weight for the ith
   observation. If any of the weights are greater than one, then they are scaled so
   that the maximum weight is one.

   prior
   Specification of a conjugate prior on the means and variances. See the help file
   for priorControl for further information. The default assumes no prior.
control A list of control parameters for EM. The defaults are set by the call `emControl`.

Vinv If the model is to include a noise term, \( V_{inv} \) is an estimate of the reciprocal hypervolume of the data region. If set to a negative value or 0, the model will include a noise term with the reciprocal hypervolume estimated by the function `hypvol`. The default is not to assume a noise term in the model through the setting \( V_{inv}=NULL \).

warn A logical value indicating whether or not certain warnings (usually related to singularity) should be issued when the estimation fails. The default is set by `warn` using `mclust.options`.

... Catches unused arguments in indirect or list calls via `do.call`.

Value A list including the following components:

- `modelName` A character string identifying the model (same as the input argument).
- `z` A matrix whose \([i,k]\)th entry is the conditional probability of the \(i\)th observation belonging to the \(k\)th component of the mixture.
- `parameters` A vector whose \(k\)th component is the mixing proportion for the \(k\)th component of the mixture model. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.
- `mean` The mean for each component. If there is more than one component, this is a matrix whose \(k\)th column is the mean of the \(k\)th component of the mixture model.
- `variance` A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
- `Vinv` The estimate of the reciprocal hypervolume of the data region used in the computation when the input indicates the addition of a noise component to the model.
- `loglik` The log likelihood for the data in the mixture model.

Attributes: "info" Information on the iteration.
"WARNING" An appropriate warning if problems are encountered in the computations.

Author(s)

Thomas Brendan Murphy

See Also

`me, meE, ..., meVVV, em, mstep, estep, priorControl, mclustModelNames, mclustVariance, mclust.options`
Examples

## Not run:
w <- rep(1,150)
w[1] <- 0
me.weighted(modelName = "VVV", data = iris[, -5], z = unmap(iris[,5]), weights=w)
## End(Not run)

---

**meE**

*EM algorithm starting with M-step for a parameterized Gaussian mixture model*

---

### Description

Implements the EM algorithm for a parameterized Gaussian mixture model, starting with the maximization step.

### Usage

```r
meE(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meV(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meX(data, prior = NULL, warn = NULL, ...)
meEII(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meVII(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meEEI(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meVEI(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meEVI(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meVVI(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meEEE(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meVEE(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meEVE(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meVVE(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meEEV(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meVEV(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meEVV(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meVVV(data, z, prior=NULL, control=emControl(), Vinv=NULL, warn=NULL, ...)
meXII(data, prior = NULL, warn = NULL, ...)
meXXI(data, prior = NULL, warn = NULL, ...)
meXXX(data, prior = NULL, warn = NULL, ...)
```

### Arguments

- **data**  
  A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

- **z**  
  A matrix whose \([i,k]\)th entry is the conditional probability of the \(i\)th observation belonging to the \(k\)th component of the mixture.
prior Specification of a conjugate prior on the means and variances. The default assumes no prior.

control A list of control parameters for EM. The defaults are set by the call emControl().

Vinv An estimate of the reciprocal hypervolume of the data region, when the model is to include a noise term. Set to a negative value or zero if a noise term is desired, but an estimate is unavailable — in that case function hypvol will be used to obtain the estimate. The default is not to assume a noise term in the model through the setting Vinv=NULL.

warn A logical value indicating whether or not certain warnings (usually related to singularity) should be issued when the estimation fails. The default is given by mclust.options("warn").

... Catches unused arguments in indirect or list calls via do.call.

Value

A list including the following components:

modelName A character string identifying the model (same as the input argument).

z A matrix whose $[i,k]$th entry is the conditional probability of the $i$th observation belonging to the $k$th component of the mixture.

parameters pro A vector whose $k$th component is the mixing proportion for the $k$th component of the mixture model. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.

mean The mean for each component. If there is more than one component, this is a matrix whose $k$th column is the mean of the $k$th component of the mixture model.

variance A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.

Vinv The estimate of the reciprocal hypervolume of the data region used in the computation when the input indicates the addition of a noise component to the model.

loglik The log likelihood for the data in the mixture model.

Attributes: "info" Information on the iteration.
"WARNING" An appropriate warning if problems are encountered in the computations.

See Also

em, me, estep, mclust.options

Examples

meVVV(data = iris[, -5], z = unmap(iris[, 5]))
mstep

M-step for parameterized Gaussian mixture models

Description

Maximization step in the EM algorithm for parameterized Gaussian mixture models.

Usage

mstep(data, modelName, z, prior = NULL, warn = NULL, ...)

Arguments

data
A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

modelName
A character string indicating the model. The help file for mclustModelNames describes the available models.

z
A matrix whose \([i,k]\)th entry is the conditional probability of the ith observation belonging to the \(k\)th component of the mixture. In analyses involving noise, this should not include the conditional probabilities for the noise component.

prior
Specification of a conjugate prior on the means and variances. The default assumes no prior.

warn
A logical value indicating whether or not certain warnings (usually related to singularity) should be issued when the estimation fails. The default is given by mclust.options("warn").

...
Catches unused arguments in indirect or list calls via do.call.

Value

A list including the following components:

modelName
A character string identifying the model (same as the input argument).

parameters
A vector whose \(k\)th component is the mixing proportion for the \(k\)th component of the mixture model. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.

mean
The mean for each component. If there is more than one component, this is a matrix whose \(k\)th column is the mean of the \(k\)th component of the mixture model.

variance
A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.

Attributes:
"info" For those models with iterative M-steps ("VEI" and "VEV"), information on the iteration.
"WARNING" An appropriate warning if problems are encountered in the computations.
Note

This function computes the M-step only for MVN mixtures, so in analyses involving noise, the conditional probabilities input should exclude those for the noise component.

In contrast to me for the EM algorithm, computations in mstep are carried out unless failure due to overflow would occur. To impose stricter tolerances on a single mstep, use me with the itmax component of the control argument set to 1.

See Also

mstepE, ..., mstepVVV, emControl, me, estep, mclust.options.

Examples

```r
## Not run:
mstep(modelName = "VII", data = iris[, -5], z = unmap(iris[, 5]))
## End(Not run)
```

---

mstepE  

M-step for a parameterized Gaussian mixture model

Description

Maximization step in the EM algorithm for a parameterized Gaussian mixture model.

Usage

mstepE(data, z, prior = NULL, warn = NULL, ...)

mstepV(data, z, prior = NULL, warn = NULL, ...)

mstepEII(data, z, prior = NULL, warn = NULL, ...)

mstepVII(data, z, prior = NULL, warn = NULL, ...)

mstepEEI(data, z, prior = NULL, warn = NULL, ...)

mstepVEI(data, z, prior = NULL, warn = NULL, control = NULL, ...)

mstepEVII(data, z, prior = NULL, warn = NULL, ...)

mstepEEEE(data, z, prior = NULL, warn = NULL, ...)

mstepEEEV(data, z, prior = NULL, warn = NULL, control = NULL, ...)

mstepVEV(data, z, prior = NULL, warn = NULL, control = NULL, ...)

mstepVVV(data, z, prior = NULL, warn = NULL, control = NULL, ...)

mstepEVE(data, z, prior = NULL, warn = NULL, control = NULL, ...)

mstepEV(data, z, prior = NULL, warn = NULL, control = NULL, ...)

mstepVE(data, z, prior = NULL, warn = NULL, control = NULL, ...)

mstepVVE(data, z, prior = NULL, warn = NULL, control = NULL, ...)

mstepVV(data, z, prior = NULL, warn = NULL, control = NULL, ...)
```
Arguments

data A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

z A matrix whose $[i,k]$th entry is the conditional probability of the $i$th observation belonging to the $k$th component of the mixture. In analyses involving noise, this should not include the conditional probabilities for the noise component.

prior Specification of a conjugate prior on the means and variances. The default assumes no prior.

warn A logical value indicating whether or not certain warnings (usually related to singularity) should be issued when the estimation fails. The default is given by mclust.options("warn").

control Values controlling termination for models "VEI" and "VEV" that have an iterative M-step. This should be a list with components named itmax and tol. These components can be of length 1 or 2; in the latter case, mstep will use the second value, under the assumption that the first applies to an outer iteration (as in the function me). The default uses the default values from the function emControl, which sets no limit on the number of iterations, and a relative tolerance of sqrt(.Machine$double.eps) on successive iterates.

... Catches unused arguments in indirect or list calls via do.call.

Value

A list including the following components:

modelName A character string identifying the model (same as the input argument).

parameters pro A vector whose $k$th component is the mixing proportion for the $k$th component of the mixture model. If the model includes a Poisson term for noise, there should be one more mixing proportion than the number of Gaussian components.

mean The mean for each component. If there is more than one component, this is a matrix whose $k$th column is the mean of the $k$th component of the mixture model.

variance A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.

Attributes: "info" For those models with iterative M-steps ("VEI" and "VEV"), information on the iteration.

"WARNING" An appropriate warning if problems are encountered in the computations.

Note

This function computes the M-step only for MVN mixtures, so in analyses involving noise, the conditional probabilities input should exclude those for the noise component.
In contrast to `me` for the EM algorithm, computations in `mstep` are carried out unless failure due to overflow would occur. To impose stricter tolerances on a single `mstep`, use `me` with the `itmax` component of the `control` argument set to 1.

**See Also**

`mstep`, `me`, `estep`, `mclustVariance`, `priorControl`, `emControl`.

**Examples**

```r
## Not run:
mstepVII(data = iris[,-5], z = unmap(iris[,5]))
## End(Not run)
```

---

### `mvn`

**Univariate or Multivariate Normal Fit**

**Description**

Computes the mean, covariance, and log-likelihood from fitting a single Gaussian to given data (univariate or multivariate normal).

**Usage**

```r
mvn( modelName, data, prior = NULL, warn = NULL, ...)
```

**Arguments**

- `modelName` A character string representing a model name. This can be either "Spherical", "Diagonal", or "Ellipsoidal" or else "X" for one-dimensional data, "XII" for a spherical Gaussian, "XXI" for a diagonal Gaussian "XXX" for a general ellipsoidal Gaussian
- `data` A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- `prior` Specification of a conjugate prior on the means and variances. The default assumes no prior.
- `warn` A logical value indicating whether or not a warning should be issued whenever a singularity is encountered. The default is given by `mclust.options("warn")`.
- `...` Catches unused arguments in indirect or list calls via `do.call`.

---

**Description**

Computes the mean, covariance, and log-likelihood from fitting a single Gaussian to given data (univariate or multivariate normal).

**Usage**

```r
mvn( modelName, data, prior = NULL, warn = NULL, ...)
```

**Arguments**

- `modelName` A character string representing a model name. This can be either "Spherical", "Diagonal", or "Ellipsoidal" or else "X" for one-dimensional data, "XII" for a spherical Gaussian, "XXI" for a diagonal Gaussian "XXX" for a general ellipsoidal Gaussian
- `data` A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- `prior` Specification of a conjugate prior on the means and variances. The default assumes no prior.
- `warn` A logical value indicating whether or not a warning should be issued whenever a singularity is encountered. The default is given by `mclust.options("warn")`.
- `...` Catches unused arguments in indirect or list calls via `do.call`.

---

### `mvn`

**Univariate or Multivariate Normal Fit**

**Description**

Computes the mean, covariance, and log-likelihood from fitting a single Gaussian to given data (univariate or multivariate normal).

**Usage**

```r
mvn( modelName, data, prior = NULL, warn = NULL, ...)
```

**Arguments**

- `modelName` A character string representing a model name. This can be either "Spherical", "Diagonal", or "Ellipsoidal" or else "X" for one-dimensional data, "XII" for a spherical Gaussian, "XXI" for a diagonal Gaussian "XXX" for a general ellipsoidal Gaussian
- `data` A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- `prior` Specification of a conjugate prior on the means and variances. The default assumes no prior.
- `warn` A logical value indicating whether or not a warning should be issued whenever a singularity is encountered. The default is given by `mclust.options("warn")`.
- `...` Catches unused arguments in indirect or list calls via `do.call`.
Value

A list including the following components:

- **modelName**: A character string identifying the model (same as the input argument).
- **parameters**
  - **mean**: The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.
  - **variance**: A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
- **loglik**: The log likelihood for the data in the mixture model.

Attributes: "WARNING" An appropriate warning if problems are encountered in the computations.

See Also

`mvnX`, `mvnXII`, `mvnXXI`, `mvnXXX`, `mclustModelNames`

Examples

```r
n <- 1000
set.seed(0)
x <- rnorm(n, mean = -1, sd = 2)
mvn(modelName = "X", x)

mu <- c(-1, 0, 1)
set.seed(0)
x <- sweep(matrix(rnorm(n*3), n, 3) %*% (2*diag(3)),
MARGIN = 2, STATS = mu, FUN = "+")
mvn(modelName = "XII", x)
mvn(modelName = "Spherical", x)

set.seed(0)
x <- sweep(matrix(rnorm(n*3), n, 3) %*% diag(1:3),
MARGIN = 2, STATS = mu, FUN = "+")
mvn(modelName = "XXI", x)
mvn(modelName = "Diagonal", x)

Sigma <- matrix(c(9,-4,1,-4,9,4,1,4,9), 3, 3)
set.seed(0)
x <- sweep(matrix(rnorm(n*3), n, 3) %*% chol(Sigma),
MARGIN = 2, STATS = mu, FUN = "+")
mvn(modelName = "XXX", x)
mvn(modelName = "Ellipsoidal", x)
```
**mvnX**

*Univariate or Multivariate Normal Fit*

**Description**

Computes the mean, covariance, and log-likelihood from fitting a single Gaussian (univariate or multivariate normal).

**Usage**

```
mvnX(data, prior = NULL, warn = NULL, ...)  
mvnXII(data, prior = NULL, warn = NULL, ...)  
mvnXXI(data, prior = NULL, warn = NULL, ...)  
mvnXXX(data, prior = NULL, warn = NULL, ...)  
```

**Arguments**

- `data`: A numeric vector, matrix, or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- `prior`: Specification of a conjugate prior on the means and variances. The default assumes no prior.
- `warn`: A logical value indicating whether or not a warning should be issued whenever a singularity is encountered. The default is given by `mclust.options("warn")`.
- `...`: Catches unused arguments in indirect or list calls via `do.call`.

**Details**

- `mvnXII` computes the best fitting Gaussian with the covariance restricted to be a multiple of the identity.
- `mvnXXI` computes the best fitting Gaussian with the covariance restricted to be diagonal.
- `mvnXXX` computes the best fitting Gaussian with ellipsoidal (unrestricted) covariance.

**Value**

A list including the following components:

- `modelName`: A character string identifying the model (same as the input argument).
- `mean`: The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.
- `variance`: A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
- `loglik`: The log likelihood for the data in the mixture model.

**Attributes**: "WARNING" An appropriate warning if problems are encountered in the computations.
### nMclustParams

**Number of Estimated Parameters in Gaussian Mixture Models**

#### Description

Gives the number of estimated parameters for parameterizations of the Gaussian mixture model that are used in MCLUST.

#### Usage

```r
nMclustParams(modelName, d, G, noise = FALSE, equalPro = FALSE, ...)
```

#### Arguments

- **modelName**: A character string indicating the model. The help file for `mclustModelNames` describes the available models.
The dimension of the data. Not used for models in which neither the shape nor
the orientation varies.

G
The number of components in the Gaussian mixture model used to compute
loglik.

noise
A logical variable indicating whether or not the model includes an optional Poisson noise component.
n equalPro
A logical variable indicating whether or not the components in the model are assumed to be present in equal proportion.

Details
To get the total number of parameters in model, add G*d for the means and G-1 for the mixing proportions if they are unequal.

Value
The number of variance parameters in the corresponding Gaussian mixture model.

See Also
bic.nVarParams.

Examples
mapply(nMclustParams, mclust.options("emModelNames"), d = 2, G = 3)

nVarParams

<table>
<thead>
<tr>
<th>Number of Variance Parameters in Gaussian Mixture Models</th>
</tr>
</thead>
</table>

Description
Gives the number of variance parameters for parameterizations of the Gaussian mixture model that are used in MCLUST.

Usage
nVarParams(modelName, d, G, ...)

Arguments
modelName
A character string indicating the model. The help file for mclustModelNames describes the available models.
d
The dimension of the data. Not used for models in which neither the shape nor the orientation varies.
G
The number of components in the Gaussian mixture model used to compute loglik.

... Catches unused arguments in indirect or list calls via do.call.
Details

To get the total number of parameters in model, add G*d for the means and G-1 for the mixing proportions if they are unequal.

Value

The number of variance parameters in the corresponding Gaussian mixture model.

References


See Also

bic, nMclustParams.

Examples

mapply(nVarParams, mclust.options("emModelNames"), d = 2, G = 3)

partconv

Numeric Encoding of a Partitioning

Description

Converts a vector interpreted as a classification or partitioning into a numeric vector.

Usage

partconv(x, consec=TRUE)

Arguments

x

A vector interpreted as a classification or partitioning.

consec

Logical value indicating whether or not consecutive class numbers should be used.

Value

Numeric encoding of x. When consec = TRUE, the distinct values in x are numbered by the order in which they appear. When consec = FALSE, each distinct value in x is numbered by the index corresponding to its first appearance in x.
partuniq

Classifies Data According to Unique Observations

Description

Gives a one-to-one mapping from unique observations to rows of a data matrix.

Usage

partuniq(x)

Arguments

x Matrix of observations.

Value

A vector of length nrow(x) with integer entries. An observation k is assigned an integer i whenever observation i is the first row of x that is identical to observation k (note that i <= k).

See Also

partconv

Examples

set.seed(0)

mat <- data.frame(lets = sample(LETTERS[1:2], 9, TRUE), nums = sample(1:2, 9, TRUE))
mat

ans <- partuniq(mat)
ans

partconv(ans, consec=TRUE)
Description

Plot combined clusterings results: classifications corresponding to \texttt{Mclust/BIC} and to the hierarchically combined classes, "entropy plots" to help to select a number of classes, and the tree structure obtained from combining mixture components.

Usage

```r
## S3 method for class 'clustCombi'
plot(x, what = c("classification", "entropy", "tree"), ...)
```

Arguments

- `x`: Object returned by \texttt{clustCombi} function.
- `what`: Type of plot.
- `...`: Other arguments to be passed to other functions: \texttt{combiPlot}, \texttt{entPlot}, \texttt{combiTree}. Please see the corresponding documentations.

Value

Classifications are plotted with \texttt{combiPlot}, which relies on the \texttt{Mclust} plot functions. Entropy plots are plotted with \texttt{entPlot} and may help to select a number of classes: please see the article cited in the references. Tree plots are produced by \texttt{combiTree} and graph the tree structure implied by the clusters combining process.

Author(s)

J.-P. Baudry, A. E. Raftery, L. Scrucca

References


See Also

\texttt{combiPlot}, \texttt{entPlot}, \texttt{combiTree}, \texttt{clustCombi}. 
Examples

## Not run:
data(Baudry_etal_2010_JCGS_examples)

## 1D Example
output <- clustCombi(data = Test1D, G=1:15)

# plots the hierarchy of combined solutions, then some "entropy plots" which
# may help one to select the number of classes (please see the article cited
# in the references)
plot(output)

## 2D Example
output <- clustCombi(data = ex4.1)

# plots the hierarchy of combined solutions, then some "entropy plots" which
# may help one to select the number of classes (please see the article cited
# in the references)
plot(output)

## 3D Example
output <- clustCombi(data = ex4.4.2)

# plots the hierarchy of combined solutions, then some "entropy plots" which
# may help one to select the number of classes (please see the article cited
# in the references)
plot(output)

## End(Not run)

plot.densityMclust  
Plots for Mixture-Based Density Estimate

Description

Plotting methods for an object of class 'mclustDensity'. Available graphs are plot of BIC values and density for univariate and bivariate data. For higher data dimensionality a scatterplot matrix of pairwise densities is drawn.

Usage

## S3 method for class 'densityMclust'
plot(x, data = NULL, what = c("BIC", "density", "diagnostic"), ...)

plotDensityMclust1(x, data = NULL, hist.col = "lightgrey",
                  hist.border = "white", breaks = "Sturges", ...)

plotDensityMclust2(x, data = NULL, nlevels = 11, levels = NULL,
plot.densityMclust

prob = c(0.25, 0.5, 0.75),
points.pch = 1, points.col = 1, points.cex = 0.8, ...)

plotDensityMclustd(x, data = NULL, nlevels = 11, levels = NULL,
prob = c(0.25, 0.5, 0.75),
points.pch = 1, points.col = 1, points.cex = 0.8,
gap = 0.2, ...)

Arguments

x
An object of class 'mclustDensity' obtained from a call to densityMclust function.
data
Optional data points.
what
The type of graph requested:
"density" = a plot of estimated density; if data is also provided the density is plotted over data points (see Details section).
"BIC" = a plot of BIC values for the estimated models versus the number of components.
"diagnostic" = diagnostic plots (only available for the one-dimensional case, see densityMclust.diagnostic)
hist.col
The color to be used to fill the bars of the histogram.
hist.border
The color of the border around the bars of the histogram.
breaks
See the argument in function hist.
points.pch, points.col, points.cex
The character symbols, colors, and magnification to be used for plotting data points.
nlevels
An integer, the number of levels to be used in plotting contour densities.
levels
A vector of density levels at which to draw the contour lines.
prob
A vector of probability levels for computing HDR. Only used if type = "hdr" and supersede previous nlevels and levels arguments.
gap
Distance between subplots, in margin lines, for the matrix of pairwise scatterplots.
...
Additional arguments passed to surfacePlot.

Details

The function plot.densityMclust allows to obtain the plot of estimated density or the graph of BIC values for evaluated models.

If what = "density" the produced plot depends on the dimensionality of the data.

For one-dimensional data a call with no data provided produces a plot of the estimated density over a sensible range of values. If data is provided the density is over-plotted on a histogram for the observed data.

For two-dimensional data further arguments available are those accepted by the surfacePlot function. In particular, the density can be represented through "contour", "hdr", "image", and
"persp" type of graph. For type = "hdr" Highest Density Regions (HDRs) are plotted for probability levels prob. See `hdrlevels` for details.

For higher dimensionality a scatterplot matrix of pairwise projected densities is drawn.

**Author(s)**
Luca Scrucca

**See Also**
`densityMclust`, `surfacePlot`, `densityMclust.diagnostics`, `Mclust`.

**Examples**

```r
## Not run:
dens <- densityMclust(faithful$waiting)
summary(dens)
summary(dens, parameters = TRUE)
plot(dens, what = "BIC", legendArgs = list(x = "topright"))
plot(dens, what = "density", data = faithful$waiting)

dens <- densityMclust(faithful)
summary(dens)
summary(dens, parameters = TRUE)
plot(dens, what = "density", data = faithful,
     drawlabels = FALSE, points.pch = 20)
plot(dens, what = "density", type = "hdr")
plot(dens, what = "density", type = "hdr", prob = seq(0.1, 0.9, by = 0.1))
plot(dens, what = "density", type = "hdr", data = faithful)
plot(dens, what = "density", type = "persp")

dens <- densityMclust(iris[,1:4])
summary(dens, parameters = TRUE)
plot(dens, what = "density", data = iris[,1:4],
     col = "slategrey", drawlabels = FALSE, nlevels = 7)
plot(dens, what = "density", type = "hdr", data = iris[,1:4])
plot(dens, what = "density", type = "persp", col = grey(0.9))

## End(Not run)
```

---

**plot.hc**

Dendrograms for Model-based Agglomerative Hierarchical Clustering

**Description**

Display two types for dendrograms for model-based hierarchical clustering objects.
## Usage

```r
## S3 method for class 'hc'
plot(x, what=c("loglik","merge"), maxG=NULL, labels=FALSE, hang=0, ...)
```

### Arguments

- **x**: An object of class `hc`.
- **what**: A character string indicating the type of dendrogram to be displayed. Possible options are:
  - "loglik": Distances between dendrogram levels are based on the classification likelihood.
  - "merge": Distances between dendrogram levels are uniform, so that levels correspond to the number of clusters.
- **maxG**: The maximum number of clusters for the dendrogram. For `what = "merge"`, the default is the number of clusters in the initial partition. For `what = "loglik"`, the default is the minimum of the maximum number of clusters for which the classification loglikelihood can be computed in most cases, and the maximum number of clusters for which the classification likelihood increases with increasing numbers of clusters.
- **labels**: A logical variable indicating whether or not to display leaf (observation) labels for the dendrogram (row names of the data). These are likely to be useful only if the number of observations is fairly small, since otherwise the labels will be too crowded to read. The default is not to display the leaf labels.
- **hang**: For `hclust` objects, this argument is the fraction of the plot height by which labels should hang below the rest of the plot. A negative value will cause the labels to hang down from 0. Because model-based hierarchical clustering does not share all of the properties of `hclust`, the hang argument won’t work in many instances.
- **...**: Additional plotting arguments.

### Details

The plotting input does not share all of the properties of `hclust` objects, hence not all plotting arguments associated with `hclust` can be expected to work here.

### Value

A dendrogram is drawn, with distances based on either the classification likelihood or the merge level (number of clusters).

### Note

If `modelName = "E"` (univariate with equal variances) or `modelName = "EII"` (multivariate with equal spherical covariances), then the underlying model is the same as for Ward’s method for hierarchical clustering.
References


See Also

hc

Examples

data(EuroUnemployment)
hcTree <- hc(modelName = "VVV", data = EuroUnemployment)
plot(hcTree, what = "loglik")
plot(hcTree, what = "loglik", labels = TRUE)
plot(hcTree, what = "loglik", maxG = 5, labels = TRUE)
plot(hcTree, what = "merge")
plot(hcTree, what = "merge", labels = TRUE)
plot(hcTree, what = "merge", labels = TRUE, hang = 0.1)
plot(hcTree, what = "merge", labels = TRUE, hang = -1)
plot(hcTree, what = "merge", labels = TRUE, maxG = 5)

plot.Mclust

Plotting method for Mclust model-based clustering

Description

Plots for model-based clustering results, such as BIC, classification, uncertainty and density.

Usage

```r
## S3 method for class 'Mclust'
plot(x, what = c("BIC", "classification", "uncertainty", "density"),
     dimens = NULL, xlab = NULL, ylab = NULL,
     addEllipses = TRUE, main = FALSE, ...)
```

Arguments

x
Output from Mclust.

what
A string specifying the type of graph requested. Available choices are:
"BIC" plot of BIC values used for choosing the number of clusters.
"classification" = a plot showing the clustering. For data in more than two dimensions a pairs plot is produced, followed by a coordinate projection plot using specified dimens. Ellipses corresponding to covariances of mixture components are also drawn if addEllipses = TRUE.

"uncertainty" = a plot of classification uncertainty. For data in more than two dimensions a coordinate projection plot is drawn using specified dimens.

"density" = a plot of estimated density. For data in more than two dimensions a matrix of contours for coordinate projection plot is drawn using specified dimens.

If not specified, in interactive sessions a menu of choices is proposed.

dimens A vector of integers specifying the dimensions of the coordinate projections in case of "classification", "uncertainty", or "density" plots.

xlab, ylab Optional labels for the x-axis and the y-axis.

addEllipses A logical indicating whether or not to add ellipses with axes corresponding to the within-cluster covariances in case of "classification" or "uncertainty" plots.

main A logical or NULL indicating whether or not to add a title to the plot identifying the type of plot drawn.

... Other graphics parameters.

Details

For more flexibility in plotting, use mclust1Dplot, mclust2Dplot, surfacePlot, coordProj, or randProj.

See Also

Mclust, plot.mclustBIC, plot.mclustICL, mclust1Dplot, mclust2Dplot, surfacePlot, coordProj, randProj.

Examples

## Not run:
precipMclust <- Mclust(precip)
plot(precipMclust)

faithfulMclust <- Mclust(faithful)
plot(faithfulMclust)

irisMclust <- Mclust(iris[, -5])
plot(irisMclust)

## End(Not run)
### Description

Plots the BIC values returned by the `mclustBIC` function.

### Usage

```r
## S3 method for class 'mclustBIC'
plot(x, G = NULL, modelNames = NULL,
     symbols = NULL, colors = NULL,
     xlab = NULL, ylab = "BIC",
     legendArgs = list(x = "bottomright", ncol = 2, cex = 1, inset = 0.01),
     ...)
```

### Arguments

- `x` Output from `mclustBIC`.
- `G` One or more numbers of components corresponding to models fit in `x`. The default is to plot the BIC for all of the numbers of components fit.
- `modelNames` One or more model names corresponding to models fit in `x`. The default is to plot the BIC for all of the models fit.
- `symbols` Either an integer or character vector assigning a plotting symbol to each unique class in `classification`. Elements in `colors` correspond to classes in order of appearance in the sequence of observations (the order used by the function `unique`). The default is given by `mclust.options("classPlotSymbols")`.
- `colors` Either an integer or character vector assigning a color to each unique class in `classification`. Elements in `colors` correspond to classes in order of appearance in the sequence of observations (the order used by the function `unique`). The default is given by `mclust.options("classPlotColors")`.
- `xlab` Optional label for the horizontal axis of the BIC plot.
- `ylab` Label for the vertical axis of the BIC plot.
- `legendArgs` Arguments to pass to the `legend` function. Set to `NULL` for no legend.
- `...` Other graphics parameters.

### Value

A plot of the BIC values.

### See Also

- `mclustBIC`
plot.MclustBootstrap

Examples

## Not run:
plot(mclustBIC(precip), legendArgs = list(x = "bottomleft"))

plot(mclustBIC(faithful))

plot(mclustBIC(iris[, -5]))

## End(Not run)

### plot.MclustBootstrap

Plot of bootstrap distributions for mixture model parameters

#### Description

Plots the bootstrap distribution of parameters as returned by the \texttt{MclustBootstrap} function.

#### Usage

```r
## S3 method for class 'MclustBootstrap'
plot(x, what = c("pro", "mean", "var"),
     show.parest = TRUE, show.confint = TRUE,
     hist.col = "grey", hist.border = "lightgrey", breaks = "Sturges",
     col = "forestgreen", lwd = 2, lty = 3,
     xlab = NULL, xlim = NULL, ylim = NULL, ...)
```

#### Arguments

- **x**: Object returned by \texttt{MclustBootstrap}.
- **what**: Character string specifying if mixing proportions ("pro"), component means ("mean") or component variances ("var") should be drawn.
- **show.parest**: A logical specifying if the parameter estimate should be drawn as vertical line.
- **show.confint**: A logical specifying if the resampling-based confidence interval should be drawn at the bottom of the graph. Confidence level can be provided as further argument \texttt{conf.level}; see \texttt{summary.MclustBootstrap}.
- **hist.col**: The color to be used to fill the bars of the histograms.
- **hist.border**: The color of the border around the bars of the histograms.
- **breaks**: See the argument in function \texttt{hist}.
- **col, lwd, lty**: The color, line width and line type to be used to represent the estimated parameters and confidence intervals.
- **xlab**: Optional label for the horizontal axis.
- **xlim, ylim**: A two-values vector of axis range for, respectively, horizontal and vertical axis.
- **...**: Other graphics parameters.
Value

A plot for each variable/component of the selected parameters.

See Also

MclustBootstrap

Examples

```r
## Not run:
data(diabetes)
X <- diabetes[, -1]
modClust <- Mclust(X, G = 3, modelNames = "VVK")
bootClust <- MclustBootstrap(modClust, nboot = 99)
par(mfrow = c(1, 3), mar = c(4, 2, 2, 0.5))
plot(bootClust, what = "pro")
par(mfrow = c(3, 3), mar = c(4, 2, 2, 0.5))
plot(bootClust, what = "mean")
## End(Not run)
```

plot.MclustDA  

Plotting method for MclustDA discriminant analysis

Description

Plots for model-based mixture discriminant analysis results, such as scatterplot of training and test data, classification of train and test data, and errors.

Usage

```r
## S3 method for class 'MclustDA'
plot(x, what = c("scatterplot", "classification", "train&test", "error"),
     newdata, newclass, dimens = NULL,
     symbols, colors, main = NULL, ...)
```

Arguments

- **x**: An object of class 'MclustDA' resulting from a call to MclustDA.
- **what**: A string specifying the type of graph requested. Available choices are:
  - "scatterplot" = a plot of training data with points marked based on the known classification. Ellipses corresponding to covariances of mixture components are also drawn.
  - "classification" = a plot of data with points marked on based the predicted classification; if newdata is provided then the test set is shown otherwise the training set.
"train&test" = a plot of training and test data with points marked according to the type of set.

"error" = a plot of training set (or test set if newdata and newclass are provided) with misclassified points marked.

If not specified, in interactive sessions a menu of choices is proposed.

newdata A data frame or matrix for test data.
newclass A vector giving the class labels for the observations in the test data (if known).
dimens A vector of integers giving the dimensions of the desired coordinate projections for multivariate data. The default is to take all the the available dimensions for plotting.
symbols Either an integer or character vector assigning a plotting symbol to each unique class. Elements in colors correspond to classes in order of appearance in the sequence of observations (the order used by the function factor). The default is given by mclust.options("classPlotSymbols").
colors Either an integer or character vector assigning a color to each unique class in classification. Elements in colors correspond to classes in order of appearance in the sequence of observations (the order used by the function factor). The default is given by mclust.options("classPlotColors").
main A logical, a character string, or NULL (default) for the main title. If NULL or FALSE no title is added to a plot. If TRUE a default title is added identifying the type of plot drawn. If a character string is provided, this is used for the title.
... further arguments passed to or from other methods.

Details
For more flexibility in plotting, use mclust1Dplot, mclust2Dplot, surfacePlot, coordProj, or randProj.

Author(s)
Luca Scrucca

See Also
MclustDA, surfacePlot, coordProj, randProj

Examples

```r
## Not run:
odd <- seq(from = 1, to = nrow(iris), by = 2)
even <- odd + 1
X.train <- iris[odd,-5]
Class.train <- iris[odd,5]
X.test <- iris[even,-5]
Class.test <- iris[even,5]

# common EEE covariance structure (which is essentially equivalent to linear discriminant analysis)
irisMclustDA <- MclustDA(X.train, Class.train, modelType = "EDDA", modelName = "EEE")
```
# common covariance structure selected by BIC
irisMclustDA <- MclustDA(X.train, Class.train, modelType = "EDDA")
summary(irisMclustDA, parameters = TRUE)
summary(irisMclustDA, newdata = X.test, newclass = Class.test)

# general covariance structure selected by BIC
irisMclustDA <- MclustDA(X.train, Class.train)
summary(irisMclustDA, parameters = TRUE)
summary(irisMclustDA, newdata = X.test, newclass = Class.test)

plot(irisMclustDA)
plot(irisMclustDA, dimens = 3:4)
plot(irisMclustDA, dimens = 4)

plot(irisMclustDA, what = "classification")
plot(irisMclustDA, what = "classification", newdata = X.test)
plot(irisMclustDA, what = "classification", dimens = 3:4)
plot(irisMclustDA, what = "classification", newdata = X.test, dimens = 3:4)
plot(irisMclustDA, what = "classification", dimens = 4)
plot(irisMclustDA, what = "classification", dimens = 4, newdata = X.test)

plot(irisMclustDA, what = "train&test", newdata = X.test)
plot(irisMclustDA, what = "train&test", newdata = X.test, dimens = 3:4)
plot(irisMclustDA, what = "train&test", newdata = X.test, dimens = 4)

plot(irisMclustDA, what = "error")
plot(irisMclustDA, what = "error", dimens = 3:4)
plot(irisMclustDA, what = "error", dimens = 4)
plot(irisMclustDA, what = "error", newdata = X.test, newclass = Class.test)
plot(irisMclustDA, what = "error", newdata = X.test, newclass = Class.test, dimens = 3:4)
plot(irisMclustDA, what = "error", newdata = X.test, newclass = Class.test, dimens = 4)

# simulated 1D data
n <- 250
set.seed(1)
triModal <- c(rnorm(n,-5), rnorm(n,0), rnorm(n,5))
triClass <- c(rep(1,n), rep(2,n), rep(3,n))
odd <- seq(from = 1, to = length(triModal), by = 2)
even <- odd + 1
triMclustDA <- MclustDA(triModal[odd], triClass[odd])
summary(triMclustDA, parameters = TRUE)
summary(triMclustDA, newdata = triModal[even], newclass = triClass[even])
plot(triMclustDA)
plot(triMclustDA, what = "classification")
plot(triMclustDA, what = "classification", newdata = triModal[even])
plot(triMclustDA, what = "train&test", newdata = triModal[even])
plot(triMclustDA, what = "error")
plot(triMclustDA, what = "error", newdata = triModal[even], newclass = triClass[even])

# simulated 2D cross data
data(cross)
odd <- seq(from = 1, to = nrow(cross), by = 2)
even <- odd + 1
crossMclustDA <- MclustDA(cross[odd,-1], cross[odd,1])
summary(crossMclustDA, parameters = TRUE)
summary(crossMclustDA, newdata = cross[even,-1], newclass = cross[even,1])
plot(crossMclustDA)
plot(crossMclustDA, what = "classification")
plot(crossMclustDA, what = "classification", newdata = cross[even,-1])
plot(crossMclustDA, what = "train&test", newdata = cross[even,-1])
plot(crossMclustDA, what = "error")
plot(crossMclustDA, what = "error", newdata = cross[even,-1], newclass = cross[even,1])

## End(Not run)

---

**plot.MclustDR**  
*Plotting method for dimension reduction for model-based clustering and classification*

### Description

Graphs data projected onto the estimated subspace for model-based clustering and classification.

### Usage

```r
## S3 method for class 'MclustDR'
plot(x, dimens,
     what = c("scatterplot", "pairs", "contour", "classification",
              "boundaries", "density", "evalues"),
     symbols, colors, col.contour = gray(0.7), col.sep = grey(0.4),
     ngrid = 200, nlevels = 5, asp = NULL, ...)
```

### Arguments

- **x**  
  An object of class 'MclustDR' resulting from a call to `MclustDR`.

- **dimens**  
  A vector of integers giving the dimensions of the desired coordinate projections for multivariate data.

- **what**  
  The type of graph requested:

  - "scatterplot" = a two-dimensional plot of data projected onto the first two directions specified by `dimens` and with data points marked according to the corresponding mixture component. By default, the first two directions are selected for plotting.

  - "pairs" = a scatterplot matrix of data projected onto the estimated subspace and with data points marked according to the corresponding mixture component. By default, all the available directions are used, unless they have been specified by `dimens`. 
"contour" = a two-dimensional plot of data projected onto the first two directions specified by `dimens` (by default, the first two directions) with density contours for classes or clusters and data points marked according to the corresponding mixture component.

"classification" = a two-dimensional plot of data projected onto the first two directions specified by `dimens` (by default, the first two directions) with classification region and data points marked according to the corresponding mixture component.

"boundaries" = a two-dimensional plot of data projected onto the first two directions specified by `dimens` (by default, the first two directions) with uncertainty boundaries and data points marked according to the corresponding mixture component. The uncertainty is shown using a greyscale with darker regions indicating higher uncertainty.

"density" = a one-dimensional plot of estimated density for the first direction specified by `dimens` (by default, the first one). A set of box-plots for each estimated cluster or known class are also shown at the bottom of the graph.

`symbols` Either an integer or character vector assigning a plotting symbol to each unique mixture component. Elements in `colors` correspond to classes in order of appearance in the sequence of observations (the order used by the function `factor`). The default is given by `mclust.options("classPlotSymbols")`.

`colors` Either an integer or character vector assigning a color to each unique cluster or known class. Elements in `colors` correspond to classes in order of appearance in the sequence of observations (the order used by the function `factor`). The default is given by `mclust.options("classPlotColors")`.

`col.contour` The color of contours in case `what = "contour"`.

`col.sep` The color of classification boundaries in case `what = "classification"`.

`ngrid` An integer specifying the number of grid points to use in evaluating the classification regions.

`nlevels` The number of levels to use in case `what = "contour"`.

`asp` For scatterplots the \( y/x \) aspect ratio, see `plot.window`.

`...` further arguments passed to or from other methods.

**Author(s)**
Luca Scrucca

**References**

**See Also**
MclustDR
Examples

```r
## Not run:
mod <- Mclust(iris[,1:4], G = 3)
dr <- MclustDR(mod)
plot(dr, what = "evalues")
plot(dr, what = "pairs")
plot(dr, what = "scatterplot", dimens = c(1,3))
plot(dr, what = "contour")
plot(dr, what = "classification", ngrid = 200)
plot(dr, what = "boundaries", ngrid = 200)
plot(dr, what = "density")
plot(dr, what = "density", dimens = 2)

data(banknote)
da <- MclustDA(banknote[,2:7], banknote$Status, G = 1:3)
dr <- MclustDR(da)
plot(dr, what = "evalues")
plot(dr, what = "pairs")
plot(dr, what = "contour")
plot(dr, what = "contour", dimens = c(1,3))
plot(dr, what = "classification", ngrid = 200)
plot(dr, what = "boundaries", ngrid = 200)
plot(dr, what = "density")
plot(dr, what = "density", dimens = 2)

## End(Not run)
```

plot.mclustICL

### ICL Plot for Model-Based Clustering

**Description**

Plots the ICL values returned by the `mclustICL` function.

**Usage**

```r
## S3 method for class 'mclustICL'
plot(x, ylab = "ICL", ...)
```

**Arguments**

- `x` Output from `mclustICL`.
- `ylab` Label for the vertical axis of the plot.
- `...` Further arguments passed to the `plot.mclustBIC` function.

**Value**

A plot of the ICL values.
plot.MclustSSC

See Also

mclustICL

Examples

## Not run:
data(faithful)
faithful.ICL = mclustICL(faithful)
plot(faithful.ICL)

## End(Not run)

plot.MclustSSC  

Tim plot method for MclustSSC semi-supervised classification

Description

Plots for semi-supervised classification based on Gaussian finite mixture models.

Usage

## S3 method for class 'MclustSSC'
plot(x, what = c("BIC", "classification", "uncertainty"), ...)

Arguments

x  
An object of class 'MclustSSC' resulting from a call to MclustSSC.

what  
A string specifying the type of graph requested. Available choices are:
"BIC" = plot of BIC values used for model selection, i.e. for choosing the
model class covariances.
"classification" = a plot of data with points marked based on the known
and the predicted classification.
"uncertainty" = a plot of classification uncertainty.
If not specified, in interactive sessions a menu of choices is proposed.

...  
further arguments passed to or from other methods. See plot.Mclust.

Author(s)

Luca Scrucca

See Also

MclustSSC
predict.densityMclust

Density estimate of multivariate observations by Gaussian finite mixture modeling

Description

Compute density estimation for multivariate observations based on Gaussian finite mixture models estimated by densityMclust.

Usage

## S3 method for class 'densityMclust'
predict(object, newdata, what = c("dens", "cdens", "z"), logarithm = FALSE, ...)

Arguments

object an object of class 'densityMclust' resulting from a call to densityMclust.
newdata a vector, a data frame or matrix giving the data. If missing the density is computed for the input data obtained from the call to densityMclust.
what a character string specifying what to retrieve: "dens" returns a vector of values for the mixture density; "cdens" returns a matrix of component densities for each mixture component (along the columns); "z" returns a matrix of conditional probabilities of each data point to belong to a mixture component.
logarithm A logical value indicating whether or not the logarithm of the density or component densities should be returned.
... further arguments passed to or from other methods.
predict.Mclust

Value

Returns a vector or a matrix of densities evaluated at newdata depending on the argument what (see above).

Author(s)

Luca Scrucca

See Also

Mclust.

Examples

## Not run:
x <- faithful$waiting
dens <- densityMclust(x)
x0 <- seq(50, 100, by = 10)
d0 <- predict(dens, x0)
plot(dens)
points(x0, d0, pch = 20)
## End(Not run)

predict.Mclust  

*Cluster multivariate observations by Gaussian finite mixture modeling*

Description

Cluster prediction for multivariate observations based on Gaussian finite mixture models estimated by Mclust.

Usage

```r
## S3 method for class 'Mclust'
predict(object, newdata, ...)
```

Arguments

- `object`: an object of class 'Mclust' resulting from a call to Mclust.
- `newdata`: a data frame or matrix giving the data. If missing the clustering data obtained from the call to Mclust are classified.
- `...`: further arguments passed to or from other methods.
predict.MclustDA

Value

Returns a list of with the following components:

- **classification**: a factor of predicted cluster labels for newdata.
- **z**: a matrix whose $i,k$th entry is the probability that observation $i$ in newdata belongs to the $k$th cluster.

Author(s)

Luca Scrucca

See Also

Mclust.

Examples

```r
model <- Mclust(faithful)

# predict cluster for the observed data
pred <- predict(model)
str(pred)
pred$z # equal to model$z
pred$classification # equal to
plot(faithful, col = pred$classification, pch = pred$classification)

# predict cluster over a grid
grid <- apply(faithful, 2, function(x) seq(min(x), max(x), length = 50))
grid <- expand.grid(eruptions = grid[,1], waiting = grid[,2])
pred <- predict(model, grid)
plot(grid, col = mclust.options("classPlotColors")[pred$classification], pch = 15, cex = 0.5)
points(faithful, pch = model$classification)
```

predict.MclustDA

*Classify multivariate observations by Gaussian finite mixture modeling*

Description

Classify multivariate observations based on Gaussian finite mixture models estimated by MclustDA.

Usage

```r
## S3 method for class 'MclustDA'
predict(object, newdata, prop = object$prop, ...)
```
Arguments

- **object**: an object of class 'MclustDA' resulting from a call to `MclustDA`.
- **newdata**: a data frame or matrix giving the data. If missing the train data obtained from the call to `MclustDA` are classified.
- **prop**: the class proportions or prior class probabilities to belong to each class; by default, this is set at the class proportions in the training data.
- **...**: further arguments passed to or from other methods.

Value

Returns a list of with the following components:

- **classification**: a factor of predicted class labels for `newdata`.
- **z**: a matrix whose $i,k$th entry is the probability that observation $i$ in `newdata` belongs to the $k$th class.

Author(s)

Luca Scrucca

See Also

`MclustDA`.

Examples

```r
## Not run:
odd <- seq(from = 1, to = nrow(iris), by = 2)
even <- odd + 1
X.train <- iris[odd,-5]
Class.train <- iris[odd,5]
X.test <- iris[even,-5]
Class.test <- iris[even,5]
irisMclustDA <- MclustDA(X.train, Class.train)
predTrain <- predict(irisMclustDA)
predTrain
predTest <- predict(irisMclustDA, X.test)
predTest
## End(Not run)
```
predict.MclustDR

Classify multivariate observations on a dimension reduced subspace by Gaussian finite mixture modeling

Description

Classify multivariate observations on a dimension reduced subspace estimated from a Gaussian finite mixture model.

Usage

## S3 method for class 'MclustDR'
predict(object, dim = 1:object$numdir, newdata, eval.points, ...)

Arguments

- **object**: an object of class 'MclustDR' resulting from a call to MclustDR.
- **dim**: the dimensions of the reduced subspace used for prediction.
- **newdata**: a data frame or matrix giving the data. If missing the data obtained from the call to MclustDR are used.
- **eval.points**: a data frame or matrix giving the data projected on the reduced subspace. If provided newdata is not used.
- **...**: further arguments passed to or from other methods.

Value

Returns a list of with the following components:

- **dir**: a matrix containing the data projected onto the dim dimensions of the reduced subspace.
- **density**: densities from mixture model for each data point.
- **z**: a matrix whose [i,k]th entry is the probability that observation i in newdata belongs to the kth class.
- **uncertainty**: The uncertainty associated with the classification.
- **classification**: A vector of values giving the MAP classification.

Author(s)

Luca Scrucca

References

predict.MclustSSC

Classification of multivariate observations by semi-supervised Gaussian finite mixtures

Description

Classify multivariate observations based on Gaussian finite mixture models estimated by \texttt{MclustSSC}.

Usage

```r
## S3 method for class 'MclustSSC'
predict(object, newdata, ...)
```

Arguments

- `object` an object of class 'MclustSSC' resulting from a call to \texttt{MclustSSC}.
- `newdata` a data frame or matrix giving the data. If missing the train data obtained from the call to \texttt{MclustSSC} are classified.
- `...` further arguments passed to or from other methods.

Value

Returns a list of with the following components:

- `classification` a factor of predicted class labels for `newdata`.
- `z` a matrix whose $[i,k]$th entry is the probability that observation $i$ in `newdata` belongs to the $k$th class.

Author(s)

Luca Scrucca

Examples

```r
mod = Mclust(iris[,1:4])
dr = MclustDR(mod)
pred = predict(dr)
str(pred)

data(banknote)
mod = MclustDA(banknote[,2:7], banknote$status)
dr = MclustDR(mod)
pred = predict(dr)
str(pred)
```
priorControl

Conjugate Prior for Gaussian Mixtures.

Description

Specify a conjugate prior for Gaussian mixtures.

Usage

priorControl(functionName = "defaultPrior", ...)

Arguments

functionName

The name of the function specifying the conjugate prior. By default the function defaultPrior is used, and this can also be used as a template for alternative specification.

...  

Optional named arguments to the function specified in functionName together with their values.

Examples

```r
## Not run:
X <- iris[,1:4]
class <- iris$Species
# randomly remove class labels
set.seed(123)
class[sample(1:length(class), size = 120)] <- NA
table(class, useNA = "ifany")
clPairs(X, ifelse(is.na(class), 0, class),
         symbols = c(0, 16, 17, 18), colors = c("grey", 4, 2, 3),
         main = "Partially classified data")

# Fit semi-supervised classification model
mod_SSC <- MclustSSC(X, class)
pred_SSC <- predict(mod_SSC)
table(Predicted = pred_SSC$classification, Actual = class, useNA = "ifany")

X_new <- data.frame(Sepal.Length = c(5, 8),
             Sepal.Width = c(3.1, 4),
             Petal.Length = c(2, 5),
             Petal.Width = c(0.5, 2))
predict(mod_SSC, newdata = X_new)
## End(Not run)
```
Details

The function priorControl is used to specify a conjugate prior for EM within MCLUST. Note that, as described in defaultPrior, in the multivariate case only 10 out of 14 models may be used in conjunction with a prior, i.e. those available in MCLUST up to version 4.4.

Value

A list with the function name as the first component. The remaining components (if any) consist of a list of arguments to the function with assigned values.

References


See Also

mclustBIC, me, mstep, defaultPrior

Examples

# default prior
irisBIC <- mclustBIC(iris,-5, prior = priorControl())
summary(irisBIC, iris,-5)

# no prior on the mean; default prior on variance
irisBIC <- mclustBIC(iris,-5, prior = priorControl(shrinkage = 0))
summary(irisBIC, iris,-5)

randomOrthogonalMatrix

Random orthogonal matrix

Description

Generate a random orthogonal basis matrix of dimension (nrow x ncol) using the method in Heiberger (1978).

Usage

randomOrthogonalMatrix(nrow, ncol, n = nrow, d = ncol, seed = NULL)
Arguments

- `nrow`: the number of rows of the resulting orthogonal matrix.
- `ncol`: the number of columns of the resulting orthogonal matrix.
- `n`: deprecated. See `nrow` above.
- `d`: deprecated. See `ncol` above.
- `seed`: an optional integer argument to use in `set.seed()` for reproducibility. By default the current seed will be used. Reproducibility can also be achieved by calling `set.seed()` before calling this function.

Details

The use of arguments `n` and `d` is deprecated and they will be removed in the future.

Value

An orthogonal matrix of dimension `nrow x ncol` such that each column is orthogonal to the other and has unit length. Because of the latter, it is also called orthonormal.

References


See Also

- `coordProj`

Examples

```r
B <- randomOrthogonalMatrix(10, 3)
zapsmall(crossprod(B))
```

Description

Plots random projections given multidimensional data and parameters of an MVN mixture model for the data.
Usage

```r
randProj(data, seeds = NULL, parameters = NULL, z = NULL, 
classification = NULL, truth = NULL, uncertainty = NULL, 
what = c("classification", "error", "uncertainty"), 
quantiles = c(0.75, 0.95), 
addEllipses = TRUE, fillEllipses = mclust.options("fillEllipses"), 
symbols = NULL, colors = NULL, scale = FALSE, 
xlim = NULL, ylim = NULL, xlab = NULL, ylab = NULL, 
cex = 1, PCH = ".", main = FALSE, ...)
```

Arguments

- **data**: A numeric matrix or data frame of observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.
- **seeds**: An integer value or a vector of integer values to be used as seed for random number generation. If multiple values are provided, then each seed should produce a different projection. By default, a single seed is drawn randomly, so each call of `randProj()` produces different projections.
- **parameters**: A named list giving the parameters of an `MCLUST` model, used to produce superimposing ellipses on the plot. The relevant components are as follows:
  - `mean`: The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.
  - `variance`: A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
- **z**: A matrix in which the [i,k]th entry gives the probability of observation i belonging to the kth class. Used to compute classification and uncertainty if those arguments aren't available.
- **classification**: A numeric or character vector representing a classification of observations (rows) of data. If present argument `z` will be ignored.
- **truth**: A numeric or character vector giving a known classification of each data point. If classification or `z` is also present, this is used for displaying classification errors.
- **uncertainty**: A numeric vector of values in (0,1) giving the uncertainty of each data point. If present argument `z` will be ignored.
- **what**: Choose from one of the following three options: "classification" (default), "error", "uncertainty".
- **quantiles**: A vector of length 2 giving quantiles used in plotting uncertainty. The smallest symbols correspond to the smallest quantile (lowest uncertainty), medium-sized (open) symbols to points falling between the given quantiles, and large (filled) symbols to those in the largest quantile (highest uncertainty). The default is (0.75,0.95).
addEllipses A logical indicating whether or not to add ellipses with axes corresponding to the within-cluster covariances in case of "classification" or "uncertainty" plots.

fillEllipses A logical specifying whether or not to fill ellipses with transparent colors when addEllipses = TRUE.

symbols Either an integer or character vector assigning a plotting symbol to each unique class in classification. Elements in colors correspond to classes in order of appearance in the sequence of observations (the order used by the function unique). The default is given by mclust.options("classPlotSymbols").

colors Either an integer or character vector assigning a color to each unique class in classification. Elements in colors correspond to classes in order of appearance in the sequence of observations (the order used by the function unique). The default is given by mclust.options("classPlotColors").

scale A logical variable indicating whether or not the two chosen dimensions should be plotted on the same scale, and thus preserve the shape of the distribution. Default: scale=FALSE

xlim, ylim Optional arguments specifying bounds for the ordinate, abscissa of the plot. This may be useful for when comparing plots.

xlab, ylab Optional arguments specifying the labels for, respectively, the horizontal and vertical axis.

cex A numerical value specifying the size of the plotting symbols. The default value is 1.

PCH An argument specifying the symbol to be used when a classification has not been specified for the data. The default value is a small dot ".".

main A logical variable or NULL indicating whether or not to add a title to the plot identifying the dimensions used.

... Other graphics parameters.

Value

A plot showing a random two-dimensional projection of the data, together with the location of the mixture components, classification, uncertainty, and/or classification errors.

The function also returns an invisible list with components basis, the randomly generated basis of the projection subspace, data, a matrix of projected data, and mu and sigma the component parameters transformed to the projection subspace.

See Also

clPairs, coordProj, mclust2Dplot, mclust.options

Examples

## Not run:
est <- meVVV(iris[,-5], unmap(iris[,5]))
par(pty = "s", mfrow = c(1,1))
randProj(iris[,,-5], seeds=1:3, parameters = est$parameters, z = est$z,
Convert mixture component covariances to decomposition form.

**Description**

Converts a set of covariance matrices from representation as a 3-D array to a parameterization by eigenvalue decomposition.

**Usage**

sigma2decomp(sigma, G = NULL, tol = sqrt(.Machine$double.eps), ...)

**Arguments**

- **sigma**
  - Either a 3-D array whose [,k]th component is the covariance matrix for the kth component in an MVN mixture model, or a single covariance matrix in the case that all components have the same covariance.
  
- **G**
  - The number of components in the mixture. When sigma is a 3-D array, the number of components can be inferred from its dimensions.
  
- **tol**
  - Tolerance for determining whether or not the covariances have equal volume, shape, and or orientation. The default is the square root of the relative machine precision, sqrt(.Machine$double.eps), which is about 1.e-8.
  
- **...**
  - Catches unused arguments from an indirect or list call via do.call.

**Value**

The covariance matrices for the mixture components in decomposition form, including the following components:

- **modelName**
  - A character string indicating the inferred model. The help file for mclustModelNames describes the available models.
  
- **d**
  - The dimension of the data.
  
- **G**
  - The number of components in the mixture model.
  
- **scale**
  - Either a G-vector giving the scale of the covariance (the dth root of its determinant) for each component in the mixture model, or a single numeric value if the scale is the same for each component.
  
- **shape**
  - Either a G by d matrix in which the kth column is the shape of the covariance matrix (normalized to have determinant 1) for the kth component, or a d-vector giving a common shape for all components.
Either a $d \times d$ by $G$ array whose $\text{[,]}, k$th entry is the orthonomal matrix whose columns are the eigenvectors of the covariance matrix of the $k$th component, or a $d \times d$ orthonormal matrix if the mixture components have a common orientation. The orientation component of `decomp` can be omitted in spherical and diagonal models, for which the principal components are parallel to the coordinate axes so that the orientation matrix is the identity.

**See Also**

decomp2sigma

**Examples**

```r
meEst <- meEEE(iris[, -5], unmap(iris[, 5]))
names(meEst$parameters$variance)
meEst$parameters$variance$Sigma

sigma2decomp(meEst$parameters$variance$Sigma, G = length(unique(iris[, 5])))
```

---

**sim**

*Simulate from Parameterized MVN Mixture Models*

**Description**

Simulate data from parameterized MVN mixture models.

**Usage**

```r
sim(modelName, parameters, n, seed = NULL, ...)
```

**Arguments**

- `modelName` A character string indicating the model. The help file for `mclustModelNames` describes the available models.
- `parameters` A list with the following components:
  - `pro` A vector whose $k$th component is the mixing proportion for the $k$th component of the mixture model. If missing, equal proportions are assumed.
  - `mean` The mean for each component. If there is more than one component, this is a matrix whose $k$th column is the mean of the $k$th component of the mixture model.
  - `variance` A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
- `n` An integer specifying the number of data points to be simulated.
- `seed` An optional integer argument to `set.seed` for reproducible random class assignment. By default the current seed will be used. Reproducibility can also be achieved by calling `set.seed` before calling `sim`.
- `...` Catches unused arguments in indirect or list calls via `do.call`.
Details

This function can be used with an indirect or list call using `do.call`, allowing the output of e.g. `mstep`, `em`, `me`, `Mclust` to be passed directly without the need to specify individual parameters as arguments.

Value

A matrix in which first column is the classification and the remaining columns are the \(n\) observations simulated from the specified MVN mixture model.

Attributes:  
- "modelName" A character string indicating the variance model used for the simulation.

See Also

`simE`, ..., `simVVV`, `Mclust`, `mstep`, `do.call`

Examples

```r
irisBIC <- mclustBIC(iris[, -5])
irisModel <- mclustModel(iris[, -5], irisBIC)
names(irisModel)
irisSim <- sim(modelName = irisModel$modelName,
               parameters = irisModel$parameters,
               n = nrow(iris))

## Not run:
do.call("sim", irisModel) # alternative call

## End(Not run)

par(pty = "s", mfrow = c(1,2))
dimnames(irisSim) <- list(NULL, c("dummy", (dimnames(iris)[[2]])[-5]))
dimens <- c(1,2)
lim1 <- apply(iris[, dimens], 2, range)
lim2 <- apply(irisSim[, dimens+1], 2, range)
lims <- apply(rbind(lim1, lim2), 2, range)
xlim <- lims[,1]
ylim <- lims[,2]

coordProj(iris[, -5], parameters=irisModel$parameters,
          classification=map(irisModel$z),
          dimens=dimens, xlim=xlim, ylim=ylim)

coordProj(iris[, -5], parameters=irisModel$parameters,
          classification=map(irisModel$z), truth = irisSim[, -1],
          dimens=dimens, xlim=xlim, ylim=ylim)

irisModel3 <- mclustModel(iris[, -5], irisBIC, G=3)
irisSim3 <- sim(modelName = irisModel3$modelName,
```
## Not run:
irisModel3$n <- NULL
irisSim3 <- do.call("sim", c(list(n=500, seed=1), irisModel3)) # alternative call
## End(Not run)
clPairs(irisSim3[, -1], cl = irisSim3[, 1])

---

Simulate from a Parameterized MVN Mixture Model

**Description**

Simulate data from a parameterized MVN mixture model.

**Usage**

```r
simE(parameters, n, seed = NULL, ...)
simV(parameters, n, seed = NULL, ...)
simEII(parameters, n, seed = NULL, ...)
simVII(parameters, n, seed = NULL, ...)
simEEI(parameters, n, seed = NULL, ...)
simVEI(parameters, n, seed = NULL, ...)
simEVI(parameters, n, seed = NULL, ...)
simVVI(parameters, n, seed = NULL, ...)
simEEE(parameters, n, seed = NULL, ...)
simVEE(parameters, n, seed = NULL, ...)
simEVE(parameters, n, seed = NULL, ...)
simVVE(parameters, n, seed = NULL, ...)
simEEV(parameters, n, seed = NULL, ...)
simVEV(parameters, n, seed = NULL, ...)
simEVV(parameters, n, seed = NULL, ...)
simVVV(parameters, n, seed = NULL, ...)
```

**Arguments**

- `parameters` A list with the following components:
  - `pro` A vector whose $k$th component is the mixing proportion for the $k$th component of the mixture model. If missing, equal proportions are assumed.
  - `mean` The mean for each component. If there is more than one component, this is a matrix whose $k$th column is the mean of the $k$th component of the mixture model.
  - `variance` A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for `mclustVariance` for details.
- `n` An integer specifying the number of data points to be simulated.
seed  An optional integer argument to set.seed() for reproducible random class assignment. By default the current seed will be used. Reproducibility can also be achieved by calling set.seed before calling sim.

... Catches unused arguments in indirect or list calls via do.call.

Details

This function can be used with an indirect or list call using do.call, allowing the output of e.g. mstep, em, me, Mclust, to be passed directly without the need to specify individual parameters as arguments.

Value

A matrix in which first column is the classification and the remaining columns are the n observations simulated from the specified MVN mixture model.

Attributes:  "modelName" A character string indicating the variance model used for the simulation.

See Also

sim, Mclust, mstepE, mclustVariance.

Examples

## Not run:
d <- 2
G <- 2
scale <- 1
shape <- c(1, 9)

O1 <- diag(2)
O2 <- diag(2)[,c(2,1)]
O <- array(cbind(O1,O2), c(2,2,2))

variance <- list(d= d, G = G, scale = scale, shape = shape, orientation = O)
mu <- matrix(0, d, G) ## center at the origin
simdat <- simEEV( n = 200, parameters = list(pro=c(1,1),mean=mu,variance=variance), seed = NULL)

cl <- simdat[,1]

sigma <- array(apply(O, 3, function(x,y) crossprod(x*y),
      y = sqrt(scale*shape)), c(2,2,2))
paramList <- list(mu = mu, sigma = sigma)
coordProj( simdat, paramList = paramList, classification = cl)

## End(Not run)
Summary method for class "Mclust".

Usage

## S3 method for class 'Mclust'
summary(object, classification = TRUE, parameters = FALSE, ...)
## S3 method for class 'summary.Mclust'
print(x, digits =getOption("digits"), ...)

Arguments

- object: An object of class 'Mclust' resulting of a call to \texttt{Mclust} or \texttt{densityMclust}.
- x: An object of class 'summary.Mclust', usually, a result of a call to \texttt{summary.Mclust}.
- classification: Logical; if TRUE a table of MAP classification/clustering of observations is printed.
- parameters: Logical; if TRUE, the parameters of mixture components are printed.
- digits: The number of significant digits to use when printing.
- ...: Further arguments passed to or from other methods.

Author(s)

Luca Scrucca

See Also

\texttt{Mclust}, \texttt{densityMclust}.

Examples

mod1 = Mclust(iris[,1:4])
summary(mod1)
summary(mod1, parameters = TRUE, classification = FALSE)

mod2 = densityMclust(faithful)
summary(mod2)
summary(mod2, parameters = TRUE)
summary.mclustBIC  Summary function for model-based clustering via BIC

Description

Optimal model characteristics and classification for model-based clustering via mclustBIC.

Usage

## S3 method for class 'mclustBIC'
summary(object, data, G, modelNames, ...)

Arguments

object  An 'mclustBIC' object, which is the result of applying mclustBIC to data.
data   The matrix or vector of observations used to generate ‘object’.
G       A vector of integers giving the numbers of mixture components (clusters) from which the best model according to BIC will be selected (as.character(G) must be a subset of the row names of object). The default is to select the best model for all numbers of mixture components used to obtain object.
modelNames A vector of integers giving the model parameterizations from which the best model according to BIC will be selected (as.character(model) must be a subset of the column names of object). The default is to select the best model for parameterizations used to obtain object.
...     Not used. For generic/method consistency.

Value

A list giving the optimal (according to BIC) parameters, conditional probabilities z, and log-likelihood, together with the associated classification and its uncertainty.

The details of the output components are as follows:

modelName  A character string denoting the model corresponding to the optimal BIC.
n          The number of observations in the data.
d          The dimension of the data.
G          The number of mixture components in the model corresponding to the optimal BIC.
bic        The optimal BIC value.
loglik     The log-likelihood corresponding to the optimal BIC.
parameters A list with the following components:
             pro  A vector whose kth component is the mixing proportion for the kth component of the mixture model. If missing, equal proportions are assumed.
mean  The mean for each component. If there is more than one component,
this is a matrix whose kth column is the mean of the kth component of the
mixture model.

variance  A list of variance parameters for the model. The components of this
list depend on the model specification. See the help file for \texttt{mclustVariance}
for details.

z  A matrix whose \([i,k]\)th entry is the probability that observation \(i\) in the data
belongs to the \(k\)th class.

classification map \((z)\): The classification corresponding to \(z\).

uncertainty  The uncertainty associated with the classification.

Attributes:  "bestBICvalues" Some of the best bic values for the analysis.
"prior" The prior as specified in the input.
"control" The control parameters for EM as specified in the input.
"initialization" The parameters used to initial EM for computing the maxi-
mum likelihood values used to obtain the BIC.

See Also

\texttt{mclustBIC mclustModel}

Examples

irisBIC <- mclustBIC(iris[, -5])
summary(irisBIC, iris[, -5])
summary(irisBIC, iris[, -5], G = 1:6, modelNames = c("VII", "VVI", "VVV"))
Details

For details about the procedure used to obtain the bootstrap distribution see `MclustBootstrap`.

See Also

`MclustBootstrap`.

Examples

```r
## Not run:
data(diabetes)
X = diabetes[, -1]
modClust = Mclust(X)
bootClust = MclustBootstrap(modClust)
summary(bootClust, what = "se")
summary(bootClust, what = "ci")

data(acidity)
modDens = densityMclust(acidity)
modDens = MclustBootstrap(modDens)
summary(modDens, what = "se")
summary(modDens, what = "ci")

## End(Not run)
```

### Summary

**summary.MclustDA**

**summarizing discriminant analysis based on Gaussian finite mixture modeling**

**Description**

Summary method for class "MclustDA".

**Usage**

```r
## S3 method for class 'MclustDA'
summary(object, parameters = FALSE, newdata, newclass, ...)
```

**Arguments**

- **object**: An object of class 'MclustDA' resulting from a call to `MclustDA`.
- **x**: An object of class 'summary.MclustDA', usually, a result of a call to `summary.MclustDA`.
- **parameters**: Logical; if TRUE, the parameters of mixture components are printed.
- **newdata**: A data frame or matrix giving the test data.
- **newclass**: A vector giving the class labels for the observations in the test data.
- **digits**: The number of significant digits to use when printing.
- **...**: Further arguments passed to or from other methods.
The function `summary.MclustDA` computes and returns a list of summary statistics of the estimated MclustDA or EDDA model for classification.

**Author(s)**
Luca Scrucca

**See Also**
`MclustDA, plot.MclustDA`

**Examples**
```r
mod = MclustDA(data = iris[,1:4], class = iris$Species)
summary(mod)
summary(mod, parameters = TRUE)
```

**Description**
Summary method for class "MclustDR".

**Usage**
```r
## S3 method for class 'MclustDR'
summary(object, numdir, std = FALSE, ...)

## S3 method for class 'summary.MclustDR'
print(x, digits = max(5,getOption("digits") - 3), ...)
```

**Arguments**
- `object`: An object of class 'MclustDR' resulting from a call to `MclustDR`.
- `x`: An object of class 'summary.MclustDR', usually, a result of a call to `summary.MclustDR`.
- `numdir`: An integer providing the number of basis directions to be printed.
- `std`: if TRUE the coefficients basis are scaled such that all predictors have unit standard deviation.
- `digits`: The number of significant digits to use when printing.
- `...`: Further arguments passed to or from other methods.

**Author(s)**
Luca Scrucca
See Also

MclustSSC, plot.MclustSSC

---

summary.MclustSSC

Summarizing semi-supervised classification model based on Gaussian finite mixtures

Description

Summary method for class "MclustSSC".

Usage

## S3 method for class 'MclustSSC'
summary(object, parameters = FALSE, ...)
## S3 method for class 'summary.MclustSSC'
print(x, digits = getOption("digits"), ...)

Arguments

- **object**: An object of class 'MclustSSC' resulting from a call to MclustSSC.
- **x**: An object of class 'summary.MclustSSC', usually, a result of a call to summary.MclustSSC.
- **parameters**: Logical; if TRUE, the parameters of mixture components are printed.
- **digits**: The number of significant digits to use when printing.
- **...**: Further arguments passed to or from other methods.

Value

The function summary.MclustSSC computes and returns a list of summary statistics of the estimated MclustSSC model for semi-supervised classification.

Author(s)

Luca Scrucca

See Also

MclustSSC, plot.MclustSSC.
Density or uncertainty surface for bivariate mixtures

Description

Plots a density or uncertainty surface given bivariate data and parameters of a MVN mixture model for the data.

Usage

surfacePlot(data, parameters,
what = c("density", "uncertainty"),
type = c("contour", "hdr", "image", "persp"),
transformation = c("none", "log", "sqrt"),
grids = 200, nlevels = 11, levels = NULL,
prob = c(0.25, 0.5, 0.75),
col = gray(0.7),
col.palette = function(...) hcl.colors(..., "blues", rev = TRUE),
hdr.palette = blue2grey.colors,
xlim = NULL, ylim = NULL, xlab = NULL, ylab = NULL,
main = FALSE, scale = FALSE, swapAxes = FALSE,
verbose = FALSE, ...)

Arguments

data A matrix, or data frame of bivariate observations. Categorical variables are not allowed. If a matrix or data frame, rows correspond to observations and columns correspond to variables.

parameters A named list giving the parameters of an MCLUST model, used to produce superimposing ellipses on the plot. The relevant components are as follows:
mean The mean for each component. If there is more than one component, this is a matrix whose kth column is the mean of the kth component of the mixture model.
variance A list of variance parameters for the model. The components of this list depend on the model specification. See the help file for mclustVariance for details.

what Choose from one of the following options: "density" (default), "uncertainty" indicating what to plot.
type Choose from one of the following three options: "contour" (default), "hdr", "image", and "persp" indicating the plot type.
transformation Choose from one of the following three options: "none" (default), "log", "sqrt" indicating a transformation to be applied before plotting.
grid The number of grid points (evenly spaced on each axis). The mixture density and uncertainty is computed at grid x grid points to produce the surface plot. Default: 100.
nlevels  The number of levels to use for a contour plot. Default: 11.
levels   A vector of levels at which to draw the lines in a contour plot.
prob     A vector of probability levels for computing HDR. Only used if type = "hdr"
          and supersede previous nlevels and levels arguments.
col      A string specifying the colour to be used for type = "contour" and type =
          "persp" plots.
col.palette A function which defines a palette of colours to be used for type = "image"
          plots.
hdr.palette A function which defines a palette of colours to be used for type = "hdr"
            plots.
xlim, ylim Optional argument specifying bounds for the ordinate, abscissa of the plot. This
          may be useful for when comparing plots.
xlab, ylab Optional argument specifying labels for the x-axis and y-axis.
main     A logical variable or NULL indicating whether or not to add a title to the plot
          identifying the dimensions used.
scale    A logical variable indicating whether or not the two dimensions should be plot-
          ted on the same scale, and thus preserve the shape of the distribution. The default
          is not to scale.
swapAxes A logical variable indicating whether or not the axes should be swapped for the
          plot.
verbose  A logical variable telling whether or not to print an indication that the function
          is in the process of computing values at the grid points, which typically takes
          some time to complete.
...      Other graphics parameters.

Details

For an image plot, a color scheme may need to be selected on the display device in order to view
the plot.

Value

A plots showing (a transformation of) the density or uncertainty for the given mixture model and
data.

The function also returns an invisible list with components x, y, and z in which x and y are the
values used to define the grid and z is the transformed density or uncertainty at the grid points.

References

C. Fraley and A. E. Raftery (2002). Model-based clustering, discriminant analysis, and density
Mixture Modeling for Model-Based Clustering, Classification, and Density Estimation. Technical
Report No. 597, Department of Statistics, University of Washington.
thyroid

See Also

mclust2Dplot

Examples

## Not run:
faithfulModel <- Mclust(faithful)
surfacePlot(faithful, parameters = faithfulModel$parameters,
            type = "contour", what = "density", transformation = "none",
            drawlabels = FALSE)
surfacePlot(faithful, parameters = faithfulModel$parameters,
            type = "persp", what = "density", transformation = "log")
surfacePlot(faithful, parameters = faithfulModel$parameters,
            type = "contour", what = "uncertainty", transformation = "log")

## End(Not run)

thyroid

Thyroid gland data

Description

Data on five laboratory tests administered to a sample of 215 patients. The tests are used to predict whether a patient's thyroid can be classified as euthyroidism (normal thyroid gland function), hypothyroidism (underactive thyroid not producing enough thyroid hormone) or hyperthyroidism (overactive thyroid producing and secreting excessive amounts of the free thyroid hormones T3 and/or thyroxine T4). Diagnosis of thyroid operation was based on a complete medical record, including anamnesis, scan, etc.

Usage

data(thyroid)

Format

A data frame with the following variables:

**Diagnosis**  Diagnosis of thyroid operation: Hypo, Normal, and Hyper.
**RT3U**  T3-resin uptake test (percentage).
**T4**  Total Serum thyroxin as measured by the isotopic displacement method.
**T3**  Total serum triiodothyronine as measured by radioimmuno assay.
**TSH**  Basal thyroid-stimulating hormone (TSH) as measured by radioimmuno assay.
**DTSH**  Maximal absolute difference of TSH value after injection of 200 micro grams of thyrotropin-releasing hormone as compared to the basal value.
Source

References


uncerPlot

*Uncertainty Plot for Model-Based Clustering*

Description
Displays the uncertainty in converting a conditional probability from EM to a classification in model-based clustering.

Usage
uncerPlot(z, truth, ...)

Arguments
- **z**: A matrix whose \(i,k\)th entry is the conditional probability of the \(i\)th observation belonging to the \(k\)th component of the mixture.
- **truth**: A numeric or character vector giving the true classification of the data.
- **...**: Provided to allow lists with elements other than the arguments can be passed in indirect or list calls with `do.call`.

Details
When truth is provided and the number of classes is compatible with \(z\), the function `compareClass` is used to to find best correspondence between classes in truth and \(z\).

Value
A plot of the uncertainty profile of the data, with uncertainties in increasing order of magnitude. If truth is supplied and the number of classes is the same as the number of columns of \(z\), the uncertainty of the misclassified data is marked by vertical lines on the plot.

See Also
`mclustBIC`, `em`, `me`, `mapClass`
Examples

```r
irisModel3 <- Mclust(iris[, -5], G = 3)
uncerPlot(z = irisModel3$z)
uncerPlot(z = irisModel3$z, truth = iris[, 5])
```

---

### unmap

**Indicator Variables given Classification**

**Description**

Converts a classification into a matrix of indicator variables.

**Usage**

```r
unmap(classification, groups=NULL, noise=NULL, ...)
```

**Arguments**

- `classification`: A numeric or character vector. Typically the distinct entries of this vector would represent a classification of observations in a data set.
- `groups`: A numeric or character vector indicating the groups from which `classification` is drawn. If not supplied, the default is to assumed to be the unique entries of `classification`.
- `noise`: A single numeric or character value used to indicate the value of `groups` corresponding to noise.
- `...`: Catches unused arguments in indirect or list calls via `do.call`.

**Value**

An \( n \) by \( m \) matrix of \((0,1)\) indicator variables, where \( n \) is the length of `classification` and \( m \) is the number of unique values or symbols in `classification`. Columns are labeled by the unique values in `classification`, and the \([i,j]\)th entry is 1 if `classification[i]` is the \(j\)th unique value or symbol in sorted order `classification`. If a `noise` value of symbol is designated, the corresponding indicator variables are relocated to the last column of the matrix.

**See Also**

`map`, `estep`, `me`
**Examples**

```r
z <- unmap(iris[,5])
z[1:5, ]

emEst <- me(modelName = "VVV", data = iris[-5], z = z)
emEst$z[1:5,]

map(emEst$z)
```

---

**wdbc**  
*Wisconsin diagnostic breast cancer (WDBC) data*

---

**Description**

The data set provides data for 569 patients on 30 features of the cell nuclei obtained from a digitized image of a fine needle aspirate (FNA) of a breast mass. For each patient the cancer was diagnosed as malignant or benign.

**Usage**

```r
data(wdbc)
```

**Format**

A data frame with 569 observations on the following variables:

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>ID number</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>Cancer diagnosis: M = malignant, B = benign</td>
</tr>
<tr>
<td>Radius_mean</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Texture_mean</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Perimeter_mean</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Area_mean</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Smoothness_mean</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Compactness_mean</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Concavity_mean</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Nconcave_mean</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Symmetry_mean</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Fractaldim_mean</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Radius_se</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Texture_se</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Perimeter_se</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Area_se</td>
<td>a numeric vector</td>
</tr>
<tr>
<td>Smoothness_se</td>
<td>a numeric vector</td>
</tr>
</tbody>
</table>
Details

The recorded features are:

- Radius as mean of distances from center to points on the perimeter
- Texture as standard deviation of gray-scale values
- Perimeter as cell nucleus perimeter
- Area as cell nucleus area
- Smoothness as local variation in radius lengths
- Compactness as cell nucleus compactness, perimeter^2 / area - 1
- Concavity as severity of concave portions of the contour
- Nconcave as number of concave portions of the contour
- Symmetry as cell nucleus shape
- Fractaldim as fractal dimension, "coastline approximation" - 1

For each feature the recorded values are computed from each image as <feature_name>_mean, <feature_name>_se, and <feature_name>_extreme, for the mean, the standard error, and the mean of the three largest values.

Source

UCI [http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)]

References

wreath

Data Simulated from a 14-Component Mixture

Description

A dataset consisting of 1000 observations drawn from a 14-component normal mixture in which the covariances of the components have the same size and shape but differ in orientation.

Usage

data(wreath)

References

Index

* classification
  MclustSSC, 107
  predict.MclustSSC, 148
  summary.MclustSSC, 164

* classif
  BrierScore, 10
  classPriorProbs, 18

* cluster
  adjustedRandIndex, 6
  bic, 9
  cdens, 12
  cdensE, 14
  cdfMclust, 15
  classError, 17
  clPairs, 21
  clustCombi, 22
  clustCombiOptim, 25
  combiPlot, 26
  combiTTree, 28
  combMat, 29
  coordProj, 30
  decomp2sigma, 35
  defaultPrior, 36
  dens, 38
  densityMclust, 39
  densityMclust.diagnostic, 40
  em, 44
  emControl, 46
  emE, 47
  entPlot, 49
  estep, 51
  estepE, 53
  gmmhd, 55
  hc, 59
  hce, 61
  hclass, 62
  hcrandomPairs, 63
  hypvol, 66
  icl, 67
  imputeData, 68
  imputePairs, 69
  map, 73
  mapClass, 73
  Mclust, 74
  mclust.options, 78
  mclust1Dplot, 81
  mclust2Dplot, 83
  mclustBIC, 85
  mclustBICupdate, 88
  MclustBootstrap, 89
  mclustBootstrapLRT, 91
  mclustICL, 102
  mclustLoglik, 104
  mclustModel, 104
  mclustModelNames, 106
  mclustVariance, 110
  me, 111
  me.weighted, 113
  meE, 115
  mstep, 117
  mstepE, 118
  mvn, 120
  mvnX, 122
  nMclustParams, 123
  nVarParams, 124
  partconv, 125
  partuniq, 126
  plot.clustCombi, 127
  plot.densityMclust, 128
  plot.hc, 130
  plot.Mclust, 132
  plot.mclustBIC, 134
  plot.MclustBootstrap, 135
  plot.mclustICL, 141
  priorControl, 149
  randProj, 151
  sigma2decomp, 154
  sim, 155
simE, 157
summary.Mclust, 159
summary.mclustBIC, 160
summary.MclustBootstrap, 161
surfacePlot, 165
uncerPlot, 168
unmap, 169
* datasets
  acidity, 5
  banknote, 7
  Baudry_etal_2010_JCGS_examples, 8
  chevron, 17
  cross, 33
  diabetes, 42
  EuroUnemployment, 54
  GvHD, 57
  thyroid, 167
  wdbc, 170
  wreath, 172
* density
  hdrlevels, 64
* dplot
  cdfMclust, 15
  densityMclust.diagnostic, 40
  plot.densityMclust, 128
* hplot
  combiTree, 28
* htest
  MclustBootstrap, 89
  mclustBootstrapLRT, 91
  summary.MclustBootstrap, 161
* multivariate
  covw, 32
  cv.MclustDA, 33
  logLik.Mclust, 70
  logLik.MclustDA, 71
  MclustDA, 93
  MclustDR, 96
  MclustDRsubsel, 99
  plot.MclustDA, 136
  plot.MclustDR, 139
  plot.MclustSSC, 142
  predict.densityMclust, 143
  predict.Mclust, 144
  predict.MclustDA, 145
  predict.MclustDR, 147
  summary.MclustDA, 162
* package
  mclust-package, 4
  acidity, 5
  adjustedRandIndex, 6
  arrows, 51
  banknote, 7
  Baudry_etal_2010_JCGS_examples, 8
  bic, 9, 67, 103, 124, 125
  bicEMtrain (mclust-deprecated), 78
  BrierScore, 10, 34
  cdens, 12, 15, 38
  cdensE, 13, 14
  cdensEEE (cdensE), 14
  cdensEEI (cdensE), 14
  cdensEEV (cdensE), 14
  cdensEII (cdensE), 14
  cdensEVE (cdensE), 14
  cdensEVI (cdensE), 14
  cdensEVV (cdensE), 14
  cdensV (cdensE), 14
  cdensVEE (cdensE), 14
  cdensVEI (cdensE), 14
  cdensVEV (cdensE), 14
  cdensVII (cdensE), 14
  cdensVVE (cdensE), 14
  cdensVVI (cdensE), 14
  cdensVVV (cdensE), 14
  cdensVVV (cdensE), 14
  cdensXXI (cdensE), 14
  cdensXXX (cdensE), 14
  cdsfMclust, 15
  chevron, 17
  classError, 6, 17, 34, 74, 95
  classPriorProbs, 18
  clPairs, 21, 31, 82, 84, 153
  clPairsLegend (clPairs), 21
  clustCombi, 22, 25–29, 50, 127
  clustCombiOptim, 25
  combiPlot, 26, 26, 29, 50, 127
  combiTree, 28, 127
  combMat, 27, 29
  coordProj, 22, 30, 82, 84, 133, 137, 151, 153
  covw, 32
  cross, 33
  cv.MclustDA (mclust-deprecated), 78
cvEMtrain (mclust-deprecated), 78

cvMclustDA, 11, 33

decomp2sigma, 35, 155
defaultPrior, 36, 75, 149, 150
dendrogram, 28
dens, 13, 15, 38, 43
densityMclust, 15, 16, 39, 41, 80, 90, 129, 130, 143, 159
densityMclust.diagnostic, 40, 129, 130
deprecated, 78
diabetes, 42
dmvnorm, 43
dnorm, 43
do.call, 13, 15, 38, 45, 54, 156

em, 44, 46, 52, 54, 73, 113, 114, 116, 168
EMclust (mclustBIC), 85
emControl, 46, 77, 80, 87, 114, 118, 120
emE, 45, 47
emEEE (emE), 47
emEEII (emE), 47
emEEV (emE), 47
deprecated, 78
diabetes, 42
dmvnorm, 43
dnorm, 43
do.call, 13, 15, 38, 45, 54, 156

dens, 13, 15, 38, 43
densityMclust, 15, 16, 39, 41, 80, 90, 129, 130, 143, 159
densityMclust.diagnostic, 40, 129, 130
deprecated, 78
diabetes, 42
dmvnorm, 43
dnorm, 43
do.call, 13, 15, 38, 45, 54, 156

em, 44, 46, 52, 54, 73, 113, 114, 116, 168
EMclust (mclustBIC), 85
emControl, 46, 77, 80, 87, 114, 118, 120
emE, 45, 47
emEEE (emE), 47
emEEII (emE), 47
emEEV (emE), 47
emEVI (emE), 47
emEV (emE), 47
deprecated, 78
diabetes, 42
dmvnorm, 43
dnorm, 43
do.call, 13, 15, 38, 45, 54, 156

estep, 13, 45, 46, 51, 54, 73, 113, 114, 116, 118, 120, 169
estepE, 52, 53
estepEEE (estepE), 53
estepEEII (estepE), 53
estepEEV (estepE), 53
estepEVI (estepE), 53
estepEV (estepE), 53
estepEVE (estepE), 53
estepVE (estepE), 53
estepV (estepE), 53
estepVII (estepE), 53
estepVIII (estepE), 53
euroUnemployment, 54
ex4.1 (Baudry_etal_2010_JCGS_examples), 8
ex4.2 (Baudry_etal_2010_JCGS_examples), 8
ex4.3 (Baudry_etal_2010_JCGS_examples), 8
ex4.4.1 (Baudry_etal_2010_JCGS_examples), 8
ex4.4.2 (Baudry_etal_2010_JCGS_examples), 8
gmmhd, 55
gmmhdClassify (gmmhd), 55
gmmhdClusterCores (gmmhd), 55
grid, 21, 41
GvHD, 57

hc, 59, 62, 63, 75, 77, 86, 87, 132
hcE, 59, 60, 61, 63
hcEEE (hcE), 61
hcEEII (hcE), 61
hcclass, 60, 62, 63, 64, 79
hcRandomPairs, 62, 63, 79
hcV (hcE), 61
hcVII (hcE), 61
hcVVV, 60, 63
hcVVV (hcE), 61
hdrlv, 64, 130
hist, 92, 129, 135
hypvol, 66, 76

icl, 67, 103
imputeData, 68, 70
imputePairs, 68, 69
matchCluster, Mclust, logLik.Mclust, logLik.MclustDA, majorityVote, map, mapClass, map, majorityVote, logLik.MclustDA, legend

INDEX

MclustSSC, mclustVariance, (MclustDRrecoverdir, MclustDRsubsel, mclustBootstrapLRT, mclustBICupdate, mclustModel, mclustModelNames, mclustSSC, mclustVariance, me. 37, 45, 46, 48, 73, 87, 111, 114, 116, 118, 120, 150, 168, 169

me.weighted, 113
meE, 113, 114, 115
meEEE (meE), 115
meEIII (meE), 115
meEEV (meE), 115
meEVE (meE), 115
meEVI (meE), 115
meVV (meE), 115
meV (meE), 115
meVEE (meE), 115
meVEI (meE), 115
meVII (meE), 115
meVEV (meE), 115
meV (meE), 115
meX (meE), 115
meXII (meE), 115
meXXI (meE), 115
meXXX (meE), 115
mstep. 15, 37, 45, 46, 48, 52, 54, 113, 114, 117, 120, 150, 156
mstepE. 118, 119, 123, 158
mstepEEE (mstepE), 118
mstepEII (mstepE), 118
mstepEVI (mstepE), 118
mstepEV (mstepE), 118
mstepV (mstepE), 118
mstepVEE (mstepE), 118
mstepVEI (mstepE), 118
mstepVII (mstepE), 118
mstepV (mstepE), 118
mstepVII (mstepE), 118
mstepV (mstepE), 118
mstepXXI (mstepE), 118
mstepXXX (mstepE), 118
mvn. 120, 123
mvnX. 121, 122
mvnXXI. 121

me. 72, 18, 73, 169
mapClass. 6, 18, 73, 168
matchCluster (imputeData), Mclust. 23, 26, 39, 40, 55, 57, 70, 71, 74, 79, 80, 88, 90, 93, 94, 97–99, 101, 103, 107, 130, 133, 144, 145, 156, 158, 159
mclust (mclust-package), 4
mclust-deprecated, 78
mclust-package, 4
mclust1Dplot, 81, 133
mclust2Dplot. 31, 82, 83, 133, 153, 167
mclustBIC. 10, 37, 46, 66, 67, 77, 79, 85, 88, 91, 93, 103–105, 107, 134, 150, 161, 168
mclustBICUpdate, 88
MclustBootstrap. 89, 135, 136, 161, 162
mclustBootstrapLR, 91, 103
MclustDA. 19, 33, 71, 80, 93, 97–99, 101, 136, 137, 145, 146, 162, 163
MclustDR. 96, 99–101, 139, 140, 147, 148, 163, 164
MclustDRRecoverdir (MclustDRsubsel), 99
MclustDRsubsel, 99
MclustDRsubsel1cycle (MclustDRsubsel), 99
MclustDRsubsel_classif (MclustDRsubsel), 99
MclustDRsubsel_cluster (MclustDRsubsel), 99
mclustICL. 67, 93, 102, 141, 142
mclustLoglik, 104
mclustModel. 87, 104, 161
mclustModelNames. 9, 10, 12, 13, 36, 38, 44, 52, 75, 77, 79, 85, 87, 90, 91, 93, 98, 99, 102, 105, 106, 108, 112–114, 117, 121, 123, 124, 154, 155
MclustSSC. 107, 142, 148, 149, 164
mclustVariance. 13–15, 30, 38, 44, 45, 48, 52–54, 76, 81, 83, 105, 109, 110,
INDEX

mvnXII (mvnX), 122
mvnXXI, 121
mvnXXI (mvnX), 122
mvnXXX, 121
mvnXXX (mvnX), 122

nMclustParams, 123, 125
nVarParams, 10, 124, 124

pairs, 21, 22, 70
partconv, 125, 126
partuniq, 126, 126
plot.clustCombi, 24, 50, 127
plot.densityMclust, 16, 40, 41, 64, 128
plot.gmmhd (gmmhd), 55
plot.hc, 60, 130
plot.Mclust, 77, 132, 142
plot.mclustBIC, 133, 134, 141
plot.MclustBootstrap, 90, 135
plot.mclustBootstrapLRT
  (mclustBootstrapLRT), 91
plot.MclustDA, 34, 95, 136, 163
plot.MclustDR, 98, 139, 164
plot.mclustICL, 103, 133, 141
plot.MclustSSC, 109, 142, 164
plot.window, 140
plotDensityMclust1
  (plot.densityMclust), 128
plotDensityMclust2
  (plot.densityMclust), 128
plotDensityMclustd
  (plot.densityMclust), 128
plotEvaluates.MclustDR (plot.MclustDR), 139
predict.densityMclust, 40, 143
predict.Mclust, 144
predict.MclustDA, 19, 34, 95, 145
predict.MclustDR, 147
predict.MclustSSC, 109, 148
predict2D.MclustDR (predict.MclustDR), 147
print.clustCombi (clustCombi), 22
print.gmmhd (gmmhd), 55
print.hc (hc), 59
print.Mclust (Mclust), 74
print.mclustBIC (mclustBIC), 85
print.MclustBootstrap
  (MclustBootstrap), 89
print.mclustBootstrapLRT
  (mclustBootstrapLRT), 91
print.MclustDA (MclustDA), 93
print.MclustDR (MclustDR), 96
print.MclustDRsubsel (MclustDRsubsel), 99
print.mclustICL (mclustICL), 102
print.mclustLoglik (mclustLoglik), 104
print.MclustSSC (MclustSSC), 107
print.summary.clustCombi (clustCombi), 22
print.summary.gmmhd (gmmhd), 55
print.summary.Mclust (summary.Mclust), 159
print.summary.mclustBIC
  (summary.mclustBIC), 160
print.summary.MclustBootstrap
  (summary.MclustBootstrap), 161
print.summary.MclustDA
  (summary.MclustDA), 162
print.summary.MclustDR
  (summary.MclustDR), 163
print.summary.mclustICL (mclustICL), 102
print.summary.MclustSSC
  (summary.MclustSSC), 164
printSummaryMclustBIC
  (summary.mclustBIC), 160
printSummaryMclustBICn
  (summary.mclustBIC), 160
priorControl, 37, 75, 77, 87, 94, 108, 113,
  114, 120, 149
quantileMclust (cdfMclust), 15
randomOrthogonalMatrix, 150
randomPairs (hcRandomPairs), 63
randProj, 31, 133, 137, 151

set.seed, 76, 86
sigma2decomp, 35, 154
sim, 155, 158
simE, 156, 157
simEEE (simE), 157
simEEI (simE), 157
simEEV (simE), 157
simEII (simE), 157
simEVE (simE), 157
simEVI (simE), 157
simEVV (simE), 157
INDEX

simV(simE), 157
simVEE (simE), 157
simVEI (simE), 157
simVEV (simE), 157
simVII (simE), 157
simVVE (simE), 157
simVVI (simE), 157
simVVV, 156
simVVV (simE), 157
summary.clustCombi (clustCombi), 22
summary.gmmhd (gmmhd), 55
summary.Mclust, 40, 77, 159
summary.mclustBIC, 87, 160
summary.MclustBootstrap, 89, 90, 135, 161
summary.MclustDA, 34, 95, 162
summary.MclustDR, 98, 163
summary.MclustDRsubsel
  (MclustDRsubsel), 99
summary.mclustICL (mclustICL), 102
summary.MclustSSC, 109, 164
summaryMclustBIC (summary.mclustBIC), 160
summaryMclustBICn (summary.mclustBIC), 160
surfacePlot, 84, 129, 130, 133, 137, 165
table, 6, 18, 74
Test1D
  (Baudry_etal_2010_JCGS_examples), 8
thyroid, 167
uncerPlot, 168
unmap, 73, 169
wdbc, 170
wreath, 172