Package ‘EMCluster’

February 19, 2015

Version 0.2-4
Date 2013-12-14
Title EM Algorithm for Model-Based Clustering of Finite Mixture Gaussian Distribution
Depends R (>= 2.14.0), MASS
LazyLoad yes
LazyData yes
Description EMCluster provides EM algorithms and several efficient initialization methods for model-based clustering of finite mixture Gaussian distribution with unstructured dispersion in both of unsupervised and semi-supervised learning.
License GPL (>= 2)
URL http://maitra.public.iastate.edu/
Author Wei-Chen Chen [aut, cre],
        Ranjan Maitra [aut],
        Volodymyr Melnykov [aut]
Maintainer Wei-Chen Chen <wccsnow@gmail.com>
NeedsCompilation yes
Repository CRAN
Date/Publication 2013-12-14 23:04:57

R topics documented:

EMCluster-package ..................................................  2
Assign Class .........................................................  3
Conversion ..........................................................  4
Dataset ...............................................................  6
EM Algorithm .........................................................  6
EM Control ...........................................................  8
Information Criteria .................................................  9
Initialization and EM ................................................. 11
EMCluster-package

Description

EMCluster provides EM algorithms and several efficient initialization methods for model-based clustering of finite mixture Gaussian distribution with unstructured dispersion in both of unsupervised and semi-supervised clustering.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>EMCluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>License:</td>
<td>GPL</td>
</tr>
<tr>
<td>LazyLoad:</td>
<td>yes</td>
</tr>
</tbody>
</table>

The install command is simply as

```r
> R CMD INSTALL EMCluster_0.2-0.tar.gz
```

from a command mode or

```r
R> install.packages("EMCluster")
```

inside an R session.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References

http://maitra.public.iastate.edu/
This function assigns cluster id to each observation in \( x \) according to the desired model \( \text{emobj} \) or specified parameters \( \pi, \mu, \) and \( \text{LTSigma} \).

### Usage

```r
assign.class(x, emobj = NULL, pi = NULL, Mu = NULL, LTSigma = NULL, lab = NULL, return.all = TRUE)
```

### Arguments

- **x**: the data matrix, dimension \( n \times p \).
- **emobj**: the desired model which is a list mainly contains \( \pi, \mu, \) and \( \text{LTSigma} \), usually a returned object from \text{init.EM}.
- **pi**: the mixing proportion, length \( K \).
- **Mu**: the centers of clusters, dimension \( K \times p \).
- **LTSigma**: the lower triangular matrices of dispersion, dimension \( K \times p(p + 1)/2 \).
- **lab**: labeled data for semi-supervised clustering, length \( n \).
- **return.all**: if returning with a whole \( \text{emobj} \) object.

### Details

The function are based either an input \( \text{emobj} \) or inputs \( \pi, \mu, \) and \( \text{LTSigma} \) to assign class id to each observation of \( x \).

If \( \text{lab} \) is submitted, then the observation with label id greater 0 will not be assigned new class.

### Value

This function returns a list containing mainly two new variables: \( \text{nc} \) (length \( K \) numbers of observations in each class) and \( \text{class} \) (length \( n \) class id).
Author(s)
Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References
http://maitra.public.iastate.edu/

See Also
initEM, emcluster.

Examples

```r
## Not run:
library(EMCluster, quiet = TRUE)
set.seed(1234)
x2 <- da2$da

et <- initEM(x2, nclass = 2)
ret.new <- assign.class(x2, ret, return.all = FALSE)
str(ret.new)

## End(Not run)
```

---

## Conversion

**Convert Matrices in Different Format**

## Description

These utility functions are to convert matrices in different formats.

## Usage

- `LTSigma2variance(x)`
- `variance2LTSigma(x)`
- `LTSigma2var(x1, p = NULL)`
- `var2LTSigma(x1)`
- `class2Gamma(class)`
- `Gamma2class(Gamma)`

## Arguments

- **x**
  a matrix/array to be converted, the dimension could be $K \times p(p + 1)/2$ or $p \times p \times K$.
- **x1**
  a vector/matrix to be converted, the length and dimension could be $p(p + 1)/2$ and $p \times p$.
- **p**
  dimension of matrix.
class \[\text{id of clusters for each observation, length } n.\]

\[\text{Gamma} \]
\[\text{containing posterior probabilities if normalized, otherwise containing component densities weighted by mixing proportion, dimension } n \times K.\]

**Details**

\[\text{LTSigma2variance} \] converts LTSigma format to 3D array, and variance2LTSigma is the inversion function.

\[\text{LTSigma2var} \] converts LTSigma format to a matrix, and \[\text{var2LTSigma} \] is the inversion function. Note that LTSigma is one component of LTSigma.

\[\text{class2Gamma} \] converts id to a Gamma matrix where with probability 1 for the cluster where the observation belongs to, and \[\text{Gamma2class} \] converts posterior to cluster id where largest posterior is picked for each observation.

**Value**

A vector/matrix/array is returned.

**Author(s)**

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

**References**

http://maitra.public.iastate.edu/

**See Also**

init.EM, emcluster.

**Examples**

```r
## Not run:
library(EMCluster, quiet = TRUE)
x <- da2$LTSigma
class <- da2$class

y <- LTSigma2variance(x)
str(y)
y <- variance2LTSigma(y)
str(y)
sum(x != y)

Gamma <- class2Gamma(class)
class.new <- Gamma2class(Gamma)
sum(class != class.new)

## End(Not run)
```
**EM Algorithm**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Description for demonstrations</th>
</tr>
</thead>
</table>

**Description**

There are four small datasets to test and demonstrate `EMCluster`.

**Usage**

```
da1
da2
da3
myiris
```

**Format**

`da1`, `da2`, `da3` are in list, and `myiris` is in matrix.

**Details**

- `da1` has 500 observations in two dimensions `da1$d$x` and `da1$d$y`, and they are in 10 clusters given in `da1$class`.
- `da2` has 2,500 observations in two dimensions, too. The true parameters are given in `da1$pi`, `da1$Mu`, and `da1$LTSigma`. There are 40 clusters given in `da1$class` for this dataset.
- `da3` is similar to `da2`, but with lower overlaps between clusters.
- `myiris` is selected from the original Iris dataset given by R.

**Author(s)**

Wei-Chen Chen (<wccsnow@gmail.com>), Ranjan Maitra, and Volodymyr Melnykov.

**References**

[http://maitra.public.iastate.edu/](http://maitra.public.iastate.edu/)

---

**EM Algorithm**

**EM Algorithm for model-based clustering**

**Description**

These are core functions of `EMCluster` performing EM algorithm for model-based clustering of finite mixture multivariate Gaussian distribution with unstructured dispersion.
Usage

emcluster(x, emobj = NULL, pi = NULL, Mu = NULL, LTSigma = NULL, lab = NULL, EMC = .EMC, assign.class = FALSE)
shortemcluster(x, emobj = NULL, pi = NULL, Mu = NULL, LTSigma = NULL, maxiter = 100, eps = 1e-2)
simple.init(x, nclass = 1)

Arguments

x the data matrix, dimension $n \times p$.
emobj the desired model which is a list mainly contains pi, Mu, and LTSigma, usually a returned object from init.EM.
pi the mixing proportion, length $K$.
Mu the centers of clusters, dimension $K \times p$.
LTSigma the lower triangular matrices of dispersion, $K \times p(p+1)/2$.
lab labeled data for semi-supervised clustering, length $n$.
EMC the control for the EM iterations.
assign.class if assigning class id.
maxiter maximum number of iterations.
eps convergent tolerance.
nclass the desired number of clusters, $K$.

Details

The emcluster mainly performs EM iterations starting from the given parameters emobj without other initializations.

The shortemcluster performs short-EM iterations as described in init.EM.

Value

The emcluster returns an object emobj with class emret which can be used in post-process or other functions such as e.step, m.step, assign.class, em.ic, and dmixmvn.

The shortemcluster also returns an object emobj with class emret which is the best of several random initializations.

The simple.init utilizes rand.EM to obtain a simple initial.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References

http://maitra.public.iastate.edu/
EM Control

See Also

init_EM, e.step, m.step, .EMControl.

Examples

```r
## Not run:
library(EMCluster, quiet = TRUE)
set.seed(1234)
x1 <- da$da

emobj <- simple.init(x1, nclass = 10)
emobj <- shortemcluster(x1, emobj)
summary(emobj)

ret <- emcluster(x1, emobj, assign.class = TRUE)
summary(ret)

## End(Not run)
```

EM Control Generator and Controller

Description

The .EMControl generates an EM control (.EMC) controlling the options and conditions of EM algorithms, i.e. this function generate a default template. One can either modify .EMC or employ this function to control EM algorithms. By default, .EMC, .EMC.Rnd, and .EC.Rndp are three native controllers as the EMCluster is loaded.

Usage

```
.EMControl(alpha = 0.99, short.iter = 200, short.eps = 1e-2,
       fixed.iter = 1, n.candidate = 3,
       EM.iter = 1000, EM.eps = 1e-6, exhaust.iter = 5)
```

EM

EM.Rnd

EM.Rndp

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alpha</td>
<td>only used in emgroup for &quot;SVD&quot; initialization.</td>
</tr>
<tr>
<td>short.iter</td>
<td>number of short-EM steps, default = 200.</td>
</tr>
<tr>
<td>short.eps</td>
<td>tolerance of short-EM steps, default = 1e-2.</td>
</tr>
<tr>
<td>fixed.iter</td>
<td>fixed iterations of EM for &quot;RndEM&quot; initialization, default = 1.</td>
</tr>
<tr>
<td>n.candidate</td>
<td>reserved for other initialization methods (unimplemented).</td>
</tr>
<tr>
<td>EM.iter</td>
<td>maximum number of long-EM steps, default = 1000.</td>
</tr>
<tr>
<td>EM.eps</td>
<td>tolerance of long-EM steps, default = 1e-6.</td>
</tr>
<tr>
<td>exhaust.iter</td>
<td>number of iterations for &quot;exhaustEM&quot; initialization, default = 5.</td>
</tr>
</tbody>
</table>
Information Criteria

Details

exhaust.iter and fixed.iter are used to control the iterations of initialization procedures.
short.iter and short.eps are used to control the short-EM iterations.
EM.iter and EM.eps are used to control the long-EM iterations.
Moreover, short.eps and EM.eps are for checking convergence of the iterations.

Value

This function returns a list as .EMC by default.
The .EMC.Rnd is equal to .EMControl(short.eps = Inf) and usually used by the rand.EM method.
The .EMC.Rndp is equal to .EMControl(fixed.iter = 5) where each random initials run 5 EM iterations in the rand.EM method.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References

http://maitra.public.iastate.edu/

See Also

init.EM, emcluster.

Examples

## Not run:
library(emCluster, quiet = TRUE)

.EMC <- .EMControl()
.EMC.Rnd <- .EMControl(short.eps = Inf)
.EMC.Rndp <- .EMControl(fixed.iter = 5)

## End(Not run)
Information Criteria

Usage

```r
em.ic(x, emobj = NULL, pi = NULL, Mu = NULL, LTSigma = NULL, l1hdeva = NULL)
em.aic(x, emobj = NULL, pi = NULL, Mu = NULL, LTSigma = NULL)
em.bic(x, emobj = NULL, pi = NULL, Mu = NULL, LTSigma = NULL)
em.clc(x, emobj = NULL, pi = NULL, Mu = NULL, LTSigma = NULL)
em.icl(x, emobj = NULL, pi = NULL, Mu = NULL, LTSigma = NULL)
em.icl.bic(x, emobj = NULL, pi = NULL, Mu = NULL, LTSigma = NULL)
```

Arguments

- `x`: the data matrix, dimension \( n \times p \).
- `emobj`: the desired model which is a list mainly contains `pi`, `Mu`, and `LTSigma`, usually a returned object from `init.EM`.
- `pi`: the mixing proportion, length \( K \).
- `Mu`: the centers of clusters, dimension \( K \times p \).
- `LTSigma`: the lower triangular matrices of dispersion, \( K \times (p(p+1)/2) \).
- `l1hdeva`: the total log likelihood value of \( x \) given `emobj`.

Details

The `em.ic` calls all other functions to compute AIC (`em.aic`), BIC (`em.bic`), CLC (`em.clc`), ICL (`em.icl`), and ICL.BIC (`em.icl.bic`). All are useful information criteria for model selections, mainly choosing number of cluster.

Value

`em.ic` returns a list containing all other information criteria for given the data `x` and the desired model `emobj`.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References

http://maitra.public.iastate.edu/

See Also

`init.EM`.

Examples

```r
## Not run:
library(EMCluster, quiet = TRUE)
x2 <- da2$data
```
Initialization and EM

These functions perform initializations (including `em.EM` and `RndEM`) followed by the EM iterations for model-based clustering of finite mixture multivariate Gaussian distribution with unstructured dispersion in both of unsupervised and semi-supervised clusterings.

Usage

```r
init.EM(x, nclass = 1, lab = NULL, EMC = .EMC,
    stable.solution = TRUE, min.n = NULL, min.n.iter = 10,
    method = c("em.EM", "Rnd.EM"))
em.EM(x, nclass = 1, lab = NULL, EMC = .EMC,
    stable.solution = TRUE, min.n = NULL, min.n.iter = 10)
rand.EM(x, nclass = 1, lab = NULL, EMC = .EMC.Rnd,
    stable.solution = TRUE, min.n = NULL, min.n.iter = 10)
exhaust.EM(x, nclass = 1, lab = NULL,
    EMC = .EMControl(short.iter = 1, short.eps = Inf),
    method = c("em.EM", "Rnd.EM"),
    stable.solution = TRUE, min.n = NULL, min.n.iter = 10);
```

Arguments

- `x` the data matrix, dimension $n \times p$.
- `nclass` the desired number of clusters, $K$.
- `lab` labeled data for semi-supervised clustering, length $n$.
- `EMC` the control for the EM iterations.
- `stable.solution` if returning a stable solution.
- `min.n` restriction for a stable solution, the minimum number of observations for every final clusters.
- `min.n.iter` restriction for a stable solution, the minimum number of iterations for trying a stable solution.
- `method` an initialization method.
Details

The init.EM calls either em.EM if method = "em.EM" or rand.EM if method = "Rand.EM".

The em.EM has two steps: short-EM has loose convergent tolerance controlled by .EMC$short.eps and try several random initializations controlled by .EMC$short.iter, while long-EM starts from the best short-EM result (in terms of log likelihood) and run to convergence with a tight tolerance controlled by .EMC$EM.eps.

The rand.EM also has two steps: first randomly pick several random initializations controlled by .EMC$short.iter, and second starts from the best of the random result (in terms of log likelihood) and run to convergence.

The lab is only for the semi-supervised clustering, and it contains pre-labeled indices between 1 and $K$ for labeled observations. Observations with index 0 is non-labeled and has to be clustered by the EM algorithm. Indices will be assigned by the results of the EM algorithm. See demo(allinit.ss,"EMCluster") for details.

The exhaust.EM also calls the init.EM with different EMC and perform exhaust.iter times of EM algorithm with different initials. The best result is returned.

Value

These functions return an object emobj with class emret which can be used in post-process or other functions such as e.step, m.step, assign.class, em.ic, and dmixmvn.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References

http://maitra.public.iastate.edu/

See Also

emcluster, .EMControl.

Examples

```r
## Not run:
library(EMCluster, quiet = TRUE)
set.seed(1234)
x <- da1$da

ret.em <- init.EM(x, nclass = 10, method = "em.EM")
ret.Rnd <- init.EM(x, nclass = 10, method = "Rand.EM", EMC = .EMC.Rnd)

emobj <- simple.init(x, nclass = 10)
ret.init <- emcluster(x, emobj, assign.class = TRUE)

par(mfrow = c(2, 2))
plotem(ret.em, x)
plotem(ret.Rnd, x)
```
plotem(ret.init, x)

## End(Not run)

### MVN

**Density of (Mixture) Multivariate Normal Distribution**

#### Description

These functions are tools for computing density of (mixture) multivariate Gaussian distribution with unstructured dispersion.

#### Usage

- `dmvn(x, mu, LTsigma, log = FALSE)`
- `dlmvn(x, mu, LTsigma, log = TRUE)`
- `dmixmvn(x, emobj = NULL, pi = NULL, Mu = NULL, LTsigma = NULL)`
- `logL(x, emobj = NULL, pi = NULL, Mu = NULL, LTsigma = NULL)`

#### Arguments

- **x**: the data matrix, dimension \( n \times p \).
- **mu**: the centers of clusters, length \( p \).
- **LTsigma**: the lower triangular matrices of dispersion, length \( p(p + 1)/2 \).
- **log**: if logarithm returned.
- **emobj**: the desired model which is a list mainly containing \( \pi \), \( \mu \), and \( LTsigma \), usually a returned object from `init.EM`.
- **pi**: the mixing proportion, length \( K \).
- **Mu**: the centers of clusters, dimension \( K \times p \).
- **LTsigma**: the lower triangular matrices of dispersion, \( K \times p(p + 1)/2 \).

#### Details

The `dmvn` and `dlmvn` compute density and log density of multivariate distribution. The `dmixmvn` computes density of mixture multivariate distribution and is based either on an input `emobj` or inputs `pi`, `Mu`, and `LTsigma` to assign class id to each observation of `x`. The `logL` returns log likelihood of mixture multivariate distribution.

#### Value

A density value is returned.

#### Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.
Other Initializations

References

http://maitra.public.iastate.edu/

See Also

init.EM, emcluster.

Examples

## Not run:
library(EMCluster, quiet = TRUE)
x2 <- da2$da
x3 <- da3$da
emobj2 <- list(pi = da2$pi, Mu = da2$Mu, LTSigma = da2$LTSigma)
emobj3 <- list(pi = da3$pi, Mu = da3$Mu, LTSigma = da3$LTSigma)

logL(x2, emobj = emobj2)
logL(x3, emobj = emobj3)

dmixmvn2 <- dmixmvn(x2, emobj2)
dmixmvn3 <- dmixmvn(x3, emobj3)

dlmvn(da2$da[1,,], da2$Mu[1,,], da2$LTSigma[1,,])
log(dmvn(da2$da[1,,], da2$Mu[1,,], da2$LTSigma[1,,]))

## End(Not run)

Description

Two more functions with different initialization method.

Usage

starts.via.svd(x, nclass = 1, method = c("em", "kmeans"),
               EMC = .EMC)
emgroup(x, nclass = 1, EMC = .EMC)

Arguments

x          the data matrix, dimension \( n \times p \).
nclass     the desired number of clusters, \( K \).
method     method with the svd initializations.
EMC        the control for the EM iterations.
Details

The `starts.via.svd` utilizes SVD to initial parameters, and the `emgroup` runs the EM algorithm starting from the initial.

Value

The `starts.via.svd` returns an object with class `svd`, and the `emgroup` returns and object `emobj` with class `emret`.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References

http://maitra.public.iastate.edu/

See Also

`init.EM`, `EMControl`.

Examples

```r
## Not run:
library(EMCluster, quiet = TRUE)
set.seed(1234)
x1 <- da$da

emobj <- emgroup(x1, nclass = 10)
summary(emobj)

ret.0 <- starts.via.svd(x1, nclass = 10, method = "kmeans")
summary(ret.0)

## End(Not run)
```

Plot EM Results

Description

The functions plot two dimensional data for clusters.

Usage

```r
plotem(emobj, x, main = NULL, xlab = NULL, ylab = NULL, ...)
plot2d(x, emobj = NULL, k = NULL, color.pch = 1,
       append.BN = TRUE, ...)```
Arguments

- **emobj**: the desired model which is a list mainly contains $\pi$, $\mu$, and $\Sigma$, usually a returned object from `init.EM`.
- **x**: the data matrix, dimension $n \times p$.
- **main**: title of plot.
- **xlab**: label of x-axis.
- **ylab**: label of y-axis.
- **...**: other parameters to the plot.
- **k**: index for symbols.
- **color.pch**: color and style for symbols.
- **append.BN**: if appending bivariate normal ellipsoid.

Details

This a simple x-y lot.

Value

A plot is returned.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References

[http://maitra.public.iastate.edu/](http://maitra.public.iastate.edu/)

See Also

`init.EM`, `emcluster`.

Examples

```r
# Not run:
library(EMCluster, quiet = TRUE)
x1 <- da$da

ret.1 <- starts.via.svd(x1, nclass = 10, method = "em")
summary(ret.1)

plotem(ret.1, x1)

# End(Not run)
```
Description

The function plots multivariate data for clusters as the parallel coordinates plot.

Usage

plotmd(x, class = NULL, xlab = "Variables", ylab = "Data", ...)

Arguments

x the data matrix, dimension $n \times p$.
class class id for all observations.
xlab label of x-axis.
ylab label of y-axis.
... other parameters to the plot.

Details

This a simplified parallel coordinate plot.

Value

A plot is returned.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References

http://maitra.public.iastate.edu/

See Also

init.EM, emcluster.
Examples

```r
## Not run:
library(EMCluster, quiet = TRUE)
set.seed(1234)

x <- myiris
ret <- em.EM(x, nclass = 5)
plotmd(x, ret$class)

## End(Not run)
```

Description

Several classes are declared in **EMCluster**, and these are functions to print and summary objects.

Usage

```r
## S3 method for class 'emret'
print(x, digits = max(4, getOption("digits") - 3), ...)
## S3 method for class 'emret'
summary(object, ...)
## S3 method for class 'svd'
summary(object, ...)
```

Arguments

- `x`: an object with the class attributes.
- `digits`: for printing out numbers.
- `object`: an object with the class attributes.
- `...`: other possible options.

Details

These are useful functions for summarizing and debugging.

Value

The results will `cat` or print on the STDOUT by default.

Author(s)

Wei-Chen Chen &lt;wccsnow@gmail.com&gt;, Ranjan Maitra, and Volodymyr Melnykov.
Rand Index

Rand Index and Adjusted Rand Index

Description
This function returns the Rand index and the adjusted Rand index for given true class ids and predicted class ids.

Usage
RRand(trcl, prcl, lab = NULL)

Arguments
trcl true class ids.
prcl predicted class ids.
lab known ids for semi-supervised clustering.

Details
All ids, trcl and prcl, should be positive integers and started from 1 to K, and the maximums are allowed to be different.
lab used in semi-supervised clustering contains the labels which are known before clustering. It should be positive integer and started from 1 for labeled data and 0 for unlabeled data.
Single Step

Value

Return a Class RRand contains Rand index and adjusted Rand index.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References

http://maitra.public.iastate.edu/

Examples

```r
## Not run:
library(EMCluster, quiet = TRUE)

true.id <- c(1, 1, 1, 2, 2, 2, 3, 3)
pred.id <- c(2, 1, 2, 1, 1, 2, 1, 1)
label <- c(0, 0, 0, 0, 1, 0, 2, 0, 0)

RRand(true.id, pred.id)
RRand(true.id, pred.id, lab = label)

## End(Not run)
```

Description

These functions are single E- and M-step of EM algorithm for model-based clustering of finite mixture multivariate Gaussian distribution with unstructured dispersion.

Usage

```r
e.step(x, emobj = NULL, pi = NULL, Mu = NULL, LTSigma = NULL, norm = TRUE)
m.step(x, emobj = NULL, Gamma = NULL, assign.class = FALSE)
```

Arguments

- `x` the data matrix, dimension $n \times p$.
- `emobj` the desired model which is a list mainly contains `pi`, `Mu`, and `LTSigma`, usually a returned object from `init.EM`.
- `pi` the mixing proportion, length $K$.
- `Mu` the centers of clusters, dimension $K \times p$.
- `LTSigma` the lower triangular matrices of dispersion, $K \times p(p + 1)/2$. 
Single Step

norm if returning normalized Gamma.
Gamma containing posterior probabilities if normalized, otherwise containing component densities weighted by mixing proportion, dimension $n \times K$.
assign.class if assigning class id.

Details

These two functions are mainly used in debugging for development and post process after model fitting.

Value

The e.step returns a list contains Gamma, the posterior probabilities if norm=TRUE, otherwise it contains component densities. This is one E-step and Gamma is used to update emobj in the M-step next.

The m.step returns a new emobj according to the Gamma from the E-step above.

Author(s)

Wei-Chen Chen <wccsnow@gmail.com>, Ranjan Maitra, and Volodymyr Melnykov.

References

http://maitra.public.iastate.edu/

See Also

init.EM.

Examples

```r
## Not run:
library(EMCluster, quiet = TRUE)
x2 <- da$da

emobj <- list(pi = da$pi, Mu = da$Mu, LTSigma = da$LTSigma)
eobj <- e.step(x2, emobj = emobj)
emobj <- m.step(x2, emobj = eobj)
emobj
```

## End(Not run)
Index

*Topic datasets
  Dataset, 6
*Topic package
  EMCluster-package, 2
*Topic programming
  EM Algorithm, 6
  EM Control, 8
  Initialization and EM, 11
  Other Initializations, 14
*Topic summary
  Print and Summary, 18
  Rand Index, 19
*Topic tool
  Assign Class, 3
  Information Criteria, 9
  Single Step, 20
*Topic utility
  Conversion, 4
  MVN, 13
  Plot EM Results, 15
  Plot Multivariate Data, 17
  .EMC (EM Control), 8
  .EMControl, 8, 12, 15
  .EMControl (EM Control), 8

Assign Class, 3
assign.class (Assign Class), 3

class2Gamma (Conversion), 4
Conversion, 4
da1 (Dataset), 6
da2 (Dataset), 6
da3 (Dataset), 6
Dataset, 6
dlmvn (MVN), 13
dmixmvn (MVN), 13
dmvn (MVN), 13
e.step, 8

e.step (Single Step), 20
EM Algorithm, 6
EM Control, 8
em.aic (Information Criteria), 9
em.bic (Information Criteria), 9
em.clc (Information Criteria), 9
em.EM (Initialization and EM), 11
em.ic (Initialization and EM), 11
em.icl (Information Criteria), 9
EMCluster (EMCluster-package), 2
ecluster, 3–5, 9, 12, 14, 16, 17, 19
emcluster (EM Algorithm), 6
EMCluster-package, 2
emgroup (Other Initializations), 14
emobj.object (EM Algorithm), 6
eEmret class (EM Algorithm), 6
exhaust.EM (Initialization and EM), 11

Gamma2class (Conversion), 4
Information Criteria, 9
init.EM, 3–5, 8–10, 14–17, 19, 21
init.EM (Initialization and EM), 11
Initialization and EM, 11

logL (MVN), 13
LTSigma2var (Conversion), 4
LTSigma2variance (Conversion), 4
m.step, 8
m.step (Single Step), 20
MVN, 13
myiris (Dataset), 6

Other Initializations, 14

Plot EM Results, 15
Plot Multivariate Data, 17
plot2d (Plot EM Results), 15
plotem (Plot EM Results), 15
plotmd (Plot Multivariate Data), 17
<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Print and Summary</td>
<td>18</td>
</tr>
<tr>
<td>print.emret (Print and Summary)</td>
<td>18</td>
</tr>
<tr>
<td>Rand Index</td>
<td>19</td>
</tr>
<tr>
<td>rand.EM (Initialization and EM)</td>
<td>11</td>
</tr>
<tr>
<td>RRand (Rand Index)</td>
<td>19</td>
</tr>
<tr>
<td>shortemcluster (EM Algorithm)</td>
<td>6</td>
</tr>
<tr>
<td>simple.init (EM Algorithm)</td>
<td>6</td>
</tr>
<tr>
<td>Single Step</td>
<td>20</td>
</tr>
<tr>
<td>starts.via.svd</td>
<td>19</td>
</tr>
<tr>
<td>starts.via.svd (Other Initializations)</td>
<td>14</td>
</tr>
<tr>
<td>summary.emret (Print and Summary)</td>
<td>18</td>
</tr>
<tr>
<td>summary.svd (Print and Summary)</td>
<td>18</td>
</tr>
<tr>
<td>var2LTsigma (Conversion)</td>
<td>4</td>
</tr>
<tr>
<td>variance2LTsigma (Conversion)</td>
<td>4</td>
</tr>
</tbody>
</table>