Package ‘DEM’

October 12, 2022

Title The Distributed EM Algorithms in Multivariate Gaussian Mixture Models

Version 0.0.0.2

Description The distributed expectation maximization algorithms are used to solve parameters of multivariate Gaussian mixture models. The philosophy of the package is described in Guo, G. (2022) <doi:10.1080/02664763.2022.2053949>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.2

Imports mvtnorm

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Qian Wang [aut, cre],
Guangbao Guo [aut],
Guoqi Qian [aut]

Maintainer Qian Wang <waqian0715@163.com>

Depends R (>= 3.5.0)

Repository CRAN

Date/Publication 2022-05-14 07:30:06 UTC

R topics documented:

DEM1 — *The DEM1 algorithm is a divide and conquer algorithm, which is used to solve the parameter estimation of multivariate Gaussian mixture model.*

Description

The DEM1 algorithm is a divide and conquer algorithm, which is used to solve the parameter estimation of multivariate Gaussian mixture model.

Usage

```
DEM1(y, M, seed, alpha0, mu0, sigma0, i, epsilon)
```

Arguments

`y`	is a data matrix
`M`	is the number of subsets
`seed`	is the recommended way to specify seeds
`alpha0`	is the initial value of the mixing weight
`mu0`	is the initial value of the mean
`sigma0`	is the initial value of the covariance
`i`	is the number of iterations
`epsilon`	is the threshold value

Value

DEM1alpha,DEM1mu,DEM1sigma,DEM1time

Examples

```
library(mvtnorm)
alpha1= c(rep(1/4,4))
mu1=matrix(0,nrow=4,ncol=4)
for (k in 1:4){
mu1[4,]=c(runif(4,(k-1)*3,k*3))
}
sigma1=list()
for (k in 1:4){
sigma1[[k]]= diag(4)*0.1
}
y= matrix(0,nrow=200,ncol=4)
for(k in 1:4){
y[c(((k-1)*200/4+1):(k*200/4)),] = rmvnorm(200/4,mu1[k,],sigma1[[k]])
}
M=5
```

```
seed=123
alpha0= alpha1
mu0=mu1
sigma0=sigma1
i=10
epsilon=0.005
DEM1(y,M,seed,alpha0,mu0,sigma0,i,epsilon)
```

DEM2 *The DEM2 algorithm is a one-step average algorithm in distributed manner, which is used to solve the parameter estimation of multivariate Gaussian mixture model.*

Description

The DEM2 algorithm is a one-step average algorithm in distributed manner, which is used to solve the parameter estimation of multivariate Gaussian mixture model.

Usage

```
DEM2(y, M, seed, alpha0, mu0, sigma0, i, epsilon)
```

Arguments

y	is a data matrix
M	is the number of subsets
seed	is the recommended way to specify seeds
alpha0	is the initial value of the mixing weight
mu0	is the initial value of the mean
sigma0	is the initial value of the covariance
i	is the number of iterations
epsilon	is the threshold value

Value

DEM2alpha,DEM2mu,DEM2sigma,DEM2time

Examples

```
library(mvtnorm)
alpha1= c(rep(1/4,4))
mu1=matrix(0,nrow=4,ncol=4)
for (k in 1:4){
mu1[4,]=c(runif(4,(k-1)*3,k*3))
}
sigma1=list()
for (k in 1:4){
```

```
sigma1[[k]]= diag(4)*0.1
}
y= matrix(0,nrow=200,ncol=4)
for(k in 1:4){
y[c(((k-1)*200/4+1):(k*200/4)),] = rmvnorm(200/4,mu1[k,],sigma1[[k]])
}
M=5
seed=123
alpha0= alpha1
mu0=mu1
sigma0=sigma1
i=10
epsilon=0.005
DEM2(y,M,seed,alpha0,mu0,sigma0,i,epsilon)
```

DMOEM *The DMOEM is an overrelaxation algorithm in distributed manner, which is used to solve the parameter estimation of multivariate Gaussian mixture model.*

Description

The DMOEM is an overrelaxation algorithm in distributed manner, which is used to solve the parameter estimation of multivariate Gaussian mixture model.

Usage

```
DMOEM(
  y,
  M,
  seed,
  alpha0,
  mu0,
  sigma0,
  MOEMalpha0,
  MOEMmu0,
  MOEMsigma0,
  omega,
  i,
  epsilon
)
```

Arguments

y	is a data matrix
M	is the number of subsets
seed	is the recommended way to specify seeds
alpha0	is the initial value of the mixing weight under the EM algorithm

`mu0`	is the initial value of the mean under the EM algorithm
`sigma0`	is the initial value of the covariance under the EM algorithm
`MOEMalpha0`	is the initial value of the mixing weight under the MOEM algorithm
`MOEMmu0`	is the initial value of the mean under the MOEM algorithm
`MOEMsigma0`	is the initial value of the covariance under the MOEM algorithm
`omega`	is the overrelaxation factor
`i`	is the number of iterations
`epsilon`	is the threshold value

Value

DMOEMalpha,DMOEMmu,DMOEMsigma,DMOEMtime

Examples

```
library(mvtnorm)
alpha1= c(rep(1/4,4))
mu1=matrix(0,nrow=4,ncol=4)
for (k in 1:4){
mu1[4,]=c(runif(4,(k-1)*3,k*3))
}
sigma1=list()
for (k in 1:4){
sigma1[[k]]= diag(4)*0.1
}
y= matrix(0,nrow=200,ncol=4)
for(k in 1:4){
y[c(((k-1)*200/4+1):(k*200/4)),] = rmvnorm(200/4,mu1[k,],sigma1[[k]])
}
M=5
seed=123
alpha0= alpha1
mu0=mu1
sigma0=sigma1
MOEMalpha0= alpha1
MOEMmu0=mu1
MOEMsigma0=sigma1
omega=0.15
i=10
epsilon=0.005
DMOEM(y,M,seed,alpha0,mu0,sigma0,MOEMalpha0,MOEMmu0,MOEMsigma0,omega,i,epsilon)
```

`DOEM1` *The DOEM1 algorithm is an online EM algorithm in distributed manner, which is used to solve the parameter estimation of multivariate Gaussian mixture model.*

Description

The DOEM1 algorithm is an online EM algorithm in distributed manner, which is used to solve the parameter estimation of multivariate Gaussian mixture model.

Usage

```
DOEM1(y, M, seed, alpha0, mu0, sigma0, i, epsilon, a, b, c)
```

Arguments

`y`	is a data matrix
`M`	is the number of subsets
`seed`	is the recommended way to specify seeds
`alpha0`	is the initial value of the mixing weight
`mu0`	is the initial value of the mean
`sigma0`	is the initial value of the covariance
`i`	is the number of iterations
`epsilon`	is the threshold value
`a`	represents the power of the reciprocal of the step size
`b`	indicates that the M-step is not implemented for the first b data points
`c`	represents online iteration starting at 1/c of the total sample size

Value

DOEM1alpha,DOEM1mu,DOEM1sigma,DOEM1time

Examples

```
library(mvtnorm)
alpha1= c(rep(1/4,4))
mu1=matrix(0,nrow=4,ncol=4)
for (k in 1:4){
mu1[4,]=c(runif(4,(k-1)*3,k*3))
}
sigma1=list()
for (k in 1:4){
sigma1[[k]]= diag(4)*0.1
}
y= matrix(0,nrow=200,ncol=4)
for(k in 1:4){
y[c(((k-1)*200/4+1):(k*200/4)),] = rmvnorm(200/4,mu1[k,],sigma1[[k]])
}
M=2
seed=123
alpha0= alpha1
mu0=mu1
sigma0=sigma1
```

```
i=10
epsilon=0.005
a=1
b=10
c=2
DOEM1(y,M,seed,alpha0,mu0,sigma0,i,epsilon,a,b,c)
```

DOEM2 *The DOEM2 algorithm is an online EM algorithm in distributed manner, which is used to solve the parameter estimation of multivariate Gaussian mixture model.*

Description

The DOEM2 algorithm is an online EM algorithm in distributed manner, which is used to solve the parameter estimation of multivariate Gaussian mixture model.

Usage

```
DOEM2(y, M, seed, alpha0, mu0, sigma0, a, b)
```

Arguments

`y`	is a data matrix
`M`	is the number of subsets
`seed`	is the recommended way to specify seeds
`alpha0`	is the initial value of the mixing weight
`mu0`	is the initial value of the mean
`sigma0`	is the initial value of the covariance
`a`	represents the power of the reciprocal of the step size
`b`	indicates that the M-step is not implemented for the first b data points

Value

DOEM2alpha,DOEM2mu,DOEM2sigma,DOEM2time

Examples

```
library(mvtnorm)
alpha1= c(rep(1/4,4))
mu1=matrix(0,nrow=4,ncol=4)
for (k in 1:4){
mu1[4,]=c(runif(4,(k-1)*3,k*3))
}
sigma1=list()
for (k in 1:4){
sigma1[[k]]= diag(4)*0.1
```

```
}
y= matrix(0,nrow=200,ncol=4)
for(k in 1:4){
y[c(((k-1)*200/4+1):(k*200/4)),] = rmvnorm(200/4,mu1[k,],sigma1[[k]])
}
M=2
seed=123
alpha0= alpha1
mu0=mu1
sigma0=sigma1
a=1
b=10
DOEM2(y,M,seed,alpha0,mu0,sigma0,a,b)
```

EM *The EM algorithm is used to solve the parameter estimation of multivariate Gaussian mixture model.*

Description

The EM algorithm is used to solve the parameter estimation of multivariate Gaussian mixture model.

Usage

```
EM(y, alpha0, mu0, sigma0, i, epsilon)
```

Arguments

y	is a data matrix
alpha0	is the initial value of the mixing weight
mu0	is the initial value of the mean
sigma0	is the initial value of the covariance
i	is the number of iterations
epsilon	is the threshold value

Value

EMalpha,EMmu,EMsigma,EMtime

Examples

```
library(mvtnorm)
alpha1= c(rep(1/4,4))
mu1=matrix(0,nrow=4,ncol=4)
for (k in 1:4){
mu1[4,]=c(runif(4,(k-1)*3,k*3))
}
sigma1=list()
```

```
for (k in 1:4){
sigma1[[k]]= diag(4)*0.1
}
y= matrix(0,nrow=200,ncol=4)
for(k in 1:4){
y[c(((k-1)*200/4+1):(k*200/4)),] = rmvnorm(200/4,mu1[k,],sigma1[[k]])
}
alpha0= alpha1
mu0=mu1
sigma0=sigma1
i=10
epsilon=0.005
EM(y,alpha0,mu0,sigma0,i,epsilon)
```

HTRU *HTRU2*

Description

The HTRU2 data

Usage

```
data("HTRU")
```

Format

A data frame with 17898 observations on the following 9 variables.

`m1` a numeric vector

`m2` a numeric vector

`m3` a numeric vector

`m4` a numeric vector

`m5` a numeric vector

`m6` a numeric vector

`m7` a numeric vector

`m8` a numeric vector

`c` a numeric vector

Details

The HTRU2 data is mainly composed of several pulsar candidate samples, which contains 17898 data points, including the 9 variables.

Source

The HTRU2 data set is from the UCI database.

References

R. J. Lyon, HTRU2, DOI: 10.6084/m9.figshare.3080389.v1.

Examples

```
data(HTRU)
## maybe str(HTRU) ; plot(HTRU) ...
```

`magic` *Magic*

Description

The magic data

Usage

```
data("magic")
```

Format

A data frame with 19020 observations on the following 11 variables.

`fLength` a numeric vector

`fWidth` a numeric vector

`fSize` a numeric vector

`fConc` a numeric vector

`fConc1` a numeric vector

`fAsym` a numeric vector

`fM3Long` a numeric vector

`fM3Trans` a numeric vector

`fAlpha` a numeric vector

`fDist` a numeric vector

`class` a character vector

Details

The magic data set is given by MAGIC project, and described by 11 features.

Source

The magic data set is from the UCI database.

References

J. Dvorak, P. Savicky. Softening Splits in Decision Trees Using Simulated Annealing. Proceedings of ICANNGA 2007, Warsaw, Part I, LNCS 4431, pp. 721-729.

Examples

```
data(magic)
## maybe str(magic) ; plot(magic) ...
```

`Skin` *Skin segmentation*

Description

The skin segmentation data

Usage

```
data("Skin")
```

Format

A data frame with 245057 observations on the following 4 variables.

`B` a numeric vector

`G` a numeric vector

`R` a numeric vector

`C` a numeric vector

Details

The skin segmentation data is related to skin texture in face image. The total number of samples is 245057, and the feature number is 3.

Source

The skin segmentation data set is from the UCI database.

References

Rajen B. Bhatt, Gaurav Sharma, Abhinav Dhall, Santanu Chaudhury, Efficient skin region segmentation using low complexity fuzzy decision tree model, IEEE-INDICON 2009, Dec 16-18, Ahmedabad, India, pp. 1-4.

Examples

```
data(Skin)
## maybe str(Skin) ; plot(Skin) ...
```

Index