

# Package ‘emmixskew’

November 7, 2013

**Title** The EM Algorithm and Skew Distribuion Mixture

**Description** EM algorithm for Mixture of Multivariate Skew Normal/T Distributioins

**Depends** lattice,mvtnorm

**Version** 1.0.1

**Author** Kui Wang, Angus Ng and Geoff McLachlan.

**Maintainer** Kui Wang <k.wang@uq.edu.au>

**License** file LICENCE

**LazyLoad** yes

## R topics documented:

|                           |           |
|---------------------------|-----------|
| bootstrap . . . . .       | 2         |
| conplot . . . . .         | 4         |
| ddmix . . . . .           | 5         |
| ddmsn . . . . .           | 6         |
| ddmst . . . . .           | 7         |
| ddmvn . . . . .           | 8         |
| ddmvt . . . . .           | 9         |
| EmSkew . . . . .          | 10        |
| EmSkew.contours . . . . . | 13        |
| EmSkewfit . . . . .       | 15        |
| EmSkewMOD . . . . .       | 17        |
| error.rate . . . . .      | 18        |
| getcov . . . . .          | 19        |
| getICL . . . . .          | 19        |
| getSWR . . . . .          | 21        |
| initEmmix . . . . .       | 21        |
| inverse . . . . .         | 23        |
| mvt.dof . . . . .         | 24        |
| rdemmix . . . . .         | 25        |
| <b>Index</b>              | <b>28</b> |

bootstrap

*Bootstrap***Description**

The standard error analysis and the bootstrap analysis of  $-2\log(\text{Lambda})$ .

**Usage**

```
bootstrap(x, n, p, g, distr, ncov, popPAR, B=99, replace=TRUE,
itmax=1000, epsilon=1e-5)
bootstrap.noc(x, n, p, g1, g2, distr, ncov, B=99, replace=TRUE,
itmax=1000, epsilon=1e-5)
```

**Arguments**

|         |  |
|---------|--|
| n       | The number of observations   |
| p       | The dimension of data  |
| B       | The number of simulated data or replacements to be tried   |
| x       | The dataset, an n by p numeric matrix, where n is number of observations and p the dimension of data.  |
| g       | The number of components of the mixture model  |
| g1, g2  | The range of the number of components of the mixture model   |
| distr   | A three letter string indicating the type of distribution to be fit. See Details.  |
| ncov    | A small integer indicating the type of covariance structure. See Details.  |
| popPAR  | A list with components <code>pro</code> , a numeric vector of the mixing proportion of each component; <code>mu</code> , a p by g matrix with each column as its corresponding mean; <code>sigma</code> , a three dimensional p by p by g array with its jth component matrix (p,p,j) as the covariance matrix for jth component of mixture models; <code>dof</code> , a vector of degrees of freedom for each component; <code>delta</code> , a p by g matrix with its columns corresponding to skew parameter vectors. |
| replace | A logical value indicating whether replacement to be used  |
| itmax   | A big integer specifying the maximum number of iterations to apply   |
| epsilon | A small number used to stop the EM algorithm loop when the relative difference between log-likelihood at each iteration become sufficient small.   |

**Details**

The distribution type, `distr`, is one of the following values: "mvn" for a multivariate normal, "mvt" for a multivariate t-distribution, "msn" for a multivariate skew normal distribution and "mst" for a multivariate skew t-distribution.

The covariance matrix type, represented by the `ncov` parameter, may be any one of the following: `ncov=1` for a common variance, `ncov=2` for a common diagonal variance, `ncov=3` for a general variance, `ncov=4` for a diagonal variance, `ncov=5` for  $\text{sigma}(h) \cdot \text{I}(p)$  (diagonal covariance with same identical diagonal element values).

When `replace` is FALSE, parametric bootstrap is used; otherwise replacement method is used.

## Value

`bootstrap` gives standard errors. `bootstrap.noc` returns a list with components `ret`, a  $B$  by  $(g_2 - g_1)$  matrix of  $-2\log(\text{Lambda})$ , `vlk`, the loglikelihood for each  $g$  in the range of  $g_1$  to  $g_2$ , and `pvalue`, the  $p$ -values of  $g$  vs  $g+1$ . The results of fitting mixture models are stored in current working directory, which can be used via command in R: `obj <- dget("ReturnOf_g_???.ret")`.

## References

McLachlan G.J. and Krishnan T. (2008). The EM Algorithm and Extensions (2nd). New Jersey: Wiley.

McLachlan G.J. and Peel D. (2000). Finite Mixture Models. New York: Wiley.

## See Also

[EmSkew, rdemmix](#)

## Examples

```
n1=300;n2=300;n3=400;
nn <-c(n1,n2,n3)
n <- sum(nn)
p <- 2
g <- 3

sigma<-array(0,c(p,p,g))
for(h in 1:3) sigma[, ,h]<-diag(p)

mu <- cbind(c(4,-4),c(3.5,4),c( 0, 0))

# for other distributions,
#delta <- cbind(c(3,3),c(1,5),c(-3,1))
#dof <- c(3,5,5)

distr="mvn"
ncov=3

#first we generate a data set
set.seed(111) #random seed is set
dat <- rdemmix(nn,p,g,distr,mu,sigma,dof=NULL,delta=NULL)

#start from initial partition
clust<- rep(1:g,nn)
obj <- EmSkewfit1(dat,g,clust,distr,ncov,itmax=1000,epsilon=1e-5)

# do bootstrap (standard error analysis)

#\dontrun{
std <- bootstrap(dat,n,p,g,distr,ncov,obj,B=19,
replace=TRUE,itmax=1000,epsilon=1e-5)
print(std)

# do bootstrap analysis of -2log(Lambda).
```

```

# alternatively data can be input as follow,
# dat <- read.table("mydata.txt",header=TRUE)
# p   <- ncol(dat)
# n   <- nrow(dat)

lad <- bootstrap.noc(dat,n,p,2,4,distr,ncov,B=19,
replace=FALSE,itmax=1000,epsilon=1e-5)
print(lad)

# return of g=2
obj2 <- dget("ReturnOf_g_2.ret")

# return of g=3
obj3 <- dget("ReturnOf_g_3.ret")

# return of g=4
obj4 <- dget("ReturnOf_g_4.ret")

#The posterior probability matrix for (g=3) is obtained by

tau <- obj3$tau

#}

```

---

conplot

---

*Functions of Contours*


---

## Description

These functions are called by EmSkew.contours, EmSkew.filter and EmSkew.flow to plot the contours of (skew) mixture density after fitting to the data.

## Usage

```

conplot(x, y, pro, mu, sigma, dof, delta, distr, grid = 300,
nrand = 6000, levels = seq(5, 95, by = 20), col = "white")
conplot2(x, y, pro, mu, sigma, dof, delta, distr, grid = 300,
         nrand = 6000, levels = seq(5, 95, by = 20))
conplot3(x, y, pro, mu, sigma, dof, delta, modpts, distr, grid = 300,
         nrand = 10000, levels = seq(5, 95, by = 20))
mypanel2(x, y, ...)
mypanel3(x, y, ...)
mypanel4(x, y, ...)
panel.density(x, col=1, ...)

```

## Arguments

|     |   |
|-----|---|
| x   | A vector of observations on variable x.                     |
| y   | A vector of observations on variable y.                     |
| pro | A vector of mixing proportions in the (skew) mixture model. |

|                     |  |
|---------------------|--|
| <code>mu</code>     | A matrix with each column corresponding to the mean or location vector of one mixture component.   |
| <code>sigma</code>  | An array of covariance matrices for each component of the mixture distribution.  |
| <code>dof</code>    | A vector of degrees of freedom when "distr"ibution is "mvt" or "mst".  |
| <code>delta</code>  | A matrix with each column as skew parameter vector of one component when "distr"ibution is "msn" or "mst".   |
| <code>distr</code>  | A three letter string indicating component distribution, "mvn"=normal distribution, "mvt"=t-distribution, "msn"=skew normal distribution, "mst"=skew t-distribution. |
| <code>modpts</code> | The mode points.   |
| <code>grid</code>   | An integer for the number of grid points in one direction.   |
| <code>nrand</code>  | A large integer for the number of random numbers being drawn.  |
| <code>levels</code> | A vector of contour percentage levels for the plots. It should be in the range of 0 to 100.  |
| <code>col</code>    | The colour of contour lines.   |
| <code>...</code>    | other  |

### Details

In most case, users do not call this function directly, instead they call the function `emmix.flow`.

### See Also

[EmSkew.flow](#)

---

ddmix

*Density Functions of Mixture Models*


---

### Description

Calculate the density of multivariate mixture models at data points for each component

### Usage

```
ddmix( dat, n, p, g, distr, mu, sigma, dof=NULL, delta=NULL)
```

### Arguments

|                    |  |
|--------------------|--|
| <code>dat</code>   | The dataset  |
| <code>n</code>     | The total number of points   |
| <code>p</code>     | Dimension of data  |
| <code>g</code>     | The number of clusters   |
| <code>distr</code> | A three letter string indicating the distribution; "mvn" for normal, "mvt" for t distribution, "msn" for skew normal, and "mst" for skew t distribution. |
| <code>mu</code>    | A numeric mean matrix with each column corresponding to the mean   |
| <code>sigma</code> | An array of dimension (p,p,g) with first two dimensions corresponding covariance matrix of each component  |
| <code>dof</code>   | A vector of degrees of freedom for each component  |
| <code>delta</code> | A matrix with each column as skew parameter vector   |

**Value**

ddmix gives an  $n$  by  $g$  matrix of logarithm of density at each data point for each component.

**References**

- McLachlan G.J. and Krishnan T. (2008). The EM Algorithm and Extensions (2nd). New Jersey: Wiley.
- McLachlan G.J. and Peel D. (2000). Finite Mixture Models. New York: Wiley.

**See Also**

[ddmvn](#), [ddmvt](#), [ddmsn](#), [ddmst](#).

**Examples**

```
p=2
g=3

#mixing propotion of each component
pro  <- c(0.3,0.3,0.4)

#specify mean and covariance matrix for each component

sigma<-array(0,c(2,2,3))
for(h in 2:3) sigma[,h]<-diag(2)
sigma[,1]<-cbind( c(1,0),c(0,.1))

mu  <- cbind(c(4,-4),c(3.5,4),c( 0, 0))

#specify other parameters for "mvt","msn","mst"

delta <- cbind(c(3,3),c(1,5),c(-3,1))
dof   <- c(3,5,5)

#specify the distribution
distr <- "mst"

y <- c(1,2)

n=1

#then the density value at y for the mixture model is

ddmix(y, n, p, g, distr, mu, sigma, dof, delta)
```

**Description**

Density and random generation for Multivariate Skew Normal distributions with mean vector `mean`, covariance matrix `cov`, and skew parameter vector `del`.

**Usage**

```
ddmsn(dat, n, p, mean, cov, del)
rdmsn(  n, p, mean, cov, del)
```

**Arguments**

|      |  |
|------|--|
| dat  | An n by p numeric matrix, the dataset    |
| n    | An integer, the number of observations   |
| p    | An integer, the dimension of data        |
| mean | A length of p vector, the mean           |
| cov  | A p by p matrix, the covariance          |
| del  | A length of p vector, the skew parameter |

**Value**

ddmsn gives the density values; rdmsn generates the random numbers

**See Also**

[rdemmix](#), [ddmvn](#), [ddmvt](#), [ddmst](#), [rdmvn](#), [rdmvt](#), [rdmst](#).

**Examples**

```
n <- 100
p <- 2

mean <- rep(0, p)
cov <- diag(p)
del <- c(0, 1)

set.seed(3214)

y <- rdmsn( n, p, mean, cov, del)

den <- ddmsn(y, n, p, mean, cov, del)
```

**Description**

Density and random generation for Multivariate Skew t-distributions with mean vector mean, covariance matrix cov, degrees of freedom nu, and skew parameter vector del.

**Usage**

```
ddmst(dat, n, p, mean, cov, nu, del)
rdmst(  n, p, mean, cov, nu, del)
```

**Arguments**

|                   |  |
|-------------------|--|
| <code>dat</code>  | An $n$ by $p$ numeric matrix, the dataset  |
| <code>n</code>    | An integer, the number of observations     |
| <code>p</code>    | An integer, the dimension of data          |
| <code>mean</code> | A length of $p$ vector, the mean           |
| <code>cov</code>  | A $p$ by $p$ matrix, the covariance        |
| <code>nu</code>   | A positive number, the degrees of freedom  |
| <code>del</code>  | A length of $p$ vector, the skew parameter |

**Value**

`ddmst` gives the density values; `rdmst` generates the random numbers

**See Also**

[rdemmix](#), [ddmvn](#), [ddmvt](#), [ddmsn](#), [rdmvn](#), [rdmvt](#), [rdmsn](#).

**Examples**

```
n <- 100
p <- 2

mean <- rep(0,p)
cov <- diag(p)
nu <- 3
del <- c(0,1)

set.seed(3214)

y <- rdmst( n,p,mean,cov,nu,del)

den <- ddmst(y,n,p,mean,cov,nu,del)
```

---

ddmvn

---

*The Multivariate Normal Distribution*


---

**Description**

Density and random generation for Multivariate Normal distributions with mean vector `mean`, and covariance matrix `cov`.

**Usage**

```
ddmvn(dat, n, p, mean, cov)
rdmvn(      n, p, mean, cov)
```



**Arguments**

|                   |   |
|-------------------|---|
| <code>dat</code>  | An $n$ by $p$ numeric matrix, the dataset |
| <code>n</code>    | An integer, the number of observations    |
| <code>p</code>    | An integer, the dimension of data         |
| <code>mean</code> | A length of $p$ vector, the mean          |
| <code>cov</code>  | A $p$ by $p$ matrix, the covariance       |

**Value**

`ddmvn` gives the density values; `rdmvn` generates the random numbers

**See Also**

[rdemmix](#), [ddmvt](#), [ddmsn](#), [ddmst](#), [rdmvt](#), [rdmsn](#), [rdmst](#).

**Examples**

```
n <- 100
p <- 2

mean <- rep(0,p)
cov <- diag(p)

set.seed(3214)

y <- rdmvn( n,p,mean,cov)

den <- ddmvn(y,n,p,mean,cov)
```

---

ddmvt

---

*The Multivariate t-Distribution*


---

**Description**

Density and random generation for Multivariate  $t$ -distributions with mean vector `mean`, covariance matrix `cov`, and degrees of freedom `nu`.

**Usage**

```
ddmvt(dat, n, p, mean, cov, nu)
rdmvt( n, p, mean, cov, nu)
```

**Arguments**

|                   |   |
|-------------------|---|
| <code>dat</code>  | An $n$ by $p$ numeric matrix, the dataset |
| <code>n</code>    | An integer, the number of observations    |
| <code>p</code>    | An integer, the dimension of data         |
| <code>mean</code> | A length of $p$ vector, the mean          |
| <code>cov</code>  | A $p$ by $p$ matrix, the covariance       |
| <code>nu</code>   | A positive number, the degrees of freedom |

**Value**

ddmvt gives the density values; rdmvt generates the random numbers

**See Also**

[rdemmix](#), [ddmvn](#), [ddmsn](#), [ddmst](#), [rdmvn](#), [rdmsn](#), [rdmst](#).

**Examples**

```
n <- 100
p <- 2

mean <- rep(0,p)
cov <- diag(p)
nu <- 3

set.seed(3214)

x <- rdmvt(n,p,mean,cov,nu)

den <- ddmvt(x,n,p,mean,cov,nu)
```

---

EmSkew

---

*The EM Algorithm and Skew Mixture Models*


---

**Description**

As a main function, EmSkew fits the data into the specified multivariate mixture models via the EM Algorithm. Distributions (univariate and multivariate) available include Normal distribution, t-distribution, Skew Normal distribution, and Skew t-distribution.

**Usage**

```
EmSkew(dat, g, distr="mvn", ncov=3, clust=NULL, init=NULL, itmax=1000,
epsilon=1e-6, nkmeans=0, nrandom=10, nhclust=FALSE, debug=TRUE,
initloop=20)
```

**Arguments**

|       |   |
|-------|---|
| dat   | The dataset, an n by p numeric matrix, where n is number of observations and p the dimension of data.                                     |
| g     | The number of components of the mixture model   |
| distr | A three letter string indicating the type of distribution to be fitted, the default value is "mvn", the Normal distribution. See Details. |
| ncov  | A small integer indicating the type of covariance structure; the default value is 3. See Details.   |
| clust | A vector of integers specifying the initial partitions of the data; the default is NULL.  |
| init  | A list containing the initial parameters for the mixture model. See details. The default value is NULL.                                   |

|                       |   |
|-----------------------|---|
| <code>itmax</code>    | A big integer specifying the maximum number of iterations to apply; the default value is 1000.  |
| <code>epsilon</code>  | A small number used to stop the EM algorithm loop when the relative difference between log-likelihood at each iteration become sufficient small; the default value is 1e-6. |
| <code>nkmeans</code>  | An integer to specify the number of KMEANS partitions to be used to find the best initial values; the default value is 0.   |
| <code>nrandom</code>  | An integer to specify the number of random partitions to be used to find the best initial values; the default value is 10.  |
| <code>nhclust</code>  | A logical value to specify whether or not to use hierarchical cluster methods; the default is FALSE. If TRUE, the Complete Linkage method will be used.                     |
| <code>debug</code>    | A logical value, if it is TRUE, the output will be printed out; FALSE silent; the default value is TRUE.  |
| <code>initloop</code> | A integer specifying the number of initial loops when searching the best initial partitions.  |

## Details

The distribution type, determined by the `distr` parameter, which may take any one of the following values: "mvn" for a multivariate normal, "mvt" for a multivariate t-distribution, "msn" for a multivariate skew normal distribution and "mst" for a multivariate skew t-distribution.

The covariance matrix type, represented by the `ncov` parameter, may be any one of the following: `ncov=1` for a common variance, `ncov=2` for a common diagonal variance, `ncov=3` for a general variance, `ncov=4` for a diagonal variance, `ncov=5` for  $\sigma(h)*I(p)$  (diagonal covariance with same identical diagonal element values).

The parameter `init` requires following elements: `pro`, a numeric vector of the mixing proportion of each component; `mu`, a  $p$  by  $g$  matrix with each column as its corresponding mean; `sigma`, a three dimensional  $p$  by  $p$  by  $g$  array with its  $j$ th component matrix  $(p,p,j)$  as the covariance matrix for  $j$ th component of mixture models; `dof`, a vector of degrees of freedom for each component; `delta`, a  $p$  by  $g$  matrix with its columns corresponding to skew parameter vectors.

Since we treat the list of `pro`, `mu`, `sigma`, `dof`, and `delta` as a common structure of parameters for our mixture models, we need to include all of them in the initial parameter list `init` by default although in some cases it does not make sense, for example, `dof` and `delta` is not applicable to normal mixture model. But in most cases, the user only need give relevant parameters in the list.

When the parameter list `init` is given, the program ignores both initial partition `clust` and automatic partition methods such as `nkmeans`; only when both `init` and `clust` are not available, the program uses automatic approaches such as k-Means partition method to find the best initial values. All three automatic approaches are used to find the best initial partition and initial values if required.

The return values include all potential parameters `pro`, `mu`, `sigma`, `dof`, and `delta`, but user should not use or interpret irrelevant information arbitrarily. For example, `dof` and `delta` for Normal mixture models.

## Value

|                    |   |
|--------------------|---|
| <code>error</code> | Error code, 0 = normal exit; 1 = did not converge within <code>itmax</code> iterations; 2 = failed to get the initial values; 3 = singularity |
| <code>aic</code>   | Akaike Information Criterion (AIC)  |
| <code>bic</code>   | Bayes Information Criterion (BIC)   |

|        |   |
|--------|---|
| ICL    | Integrated Completed Likelihood Criterion (ICL)   |
| pro    | A vector of mixing proportions.   |
| mu     | A numeric matrix with each column corresponding to the mean.  |
| sigma  | An array of dimension (p,p,g) with first two dimension corresponding covariance matrix of each component. |
| dof    | A vector of degrees of freedom for each component, see Details.   |
| delta  | A p by g matrix with each column corresponding to a skew parameter vector.                                |
| clust  | A vector of final partition   |
| loglik | The log likelihood at convergence   |
| lk     | A vector of log likelihood at each EM iteration   |
| tau    | An n by g matrix of posterior probability for each data point   |

## References

- Biernacki C. Celeux G., and Govaert G. (2000). Assessing a Mixture Model for Clustering with the integrated Completed Likelihood. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 22(7). 719-725.
- McLachlan G.J. and Krishnan T. (2008). *The EM Algorithm and Extensions* (2nd). New Jersey: Wiley.
- McLachlan G.J. and Peel D. (2000). *Finite Mixture Models*. New York: Wiley.

## See Also

[initEmmix,rdemmix.](#)

## Examples

```
#define the dimension of dataset

n1=300;n2=300;n3=400;
nn<-c(n1,n2,n3)

p  <- 2
ng <- 3

#define the parameters
sigma<-array(0,c(2,2,3))
for(h in 2:3) sigma[,h]<-diag(2)
sigma[,1]<-cbind( c(1,0.2),c(0.2,1))
mu      <- cbind(c(4,-4),c(3.5,4),c( 0, 0))

#and other parameters if required for "mvt","msn","mst"
delta  <- cbind(c(3,3),c(1,5),c(-3,1))
dof    <- c(3,5,5)

pro    <- c(0.3,0.3,0.4)

distr="mvn"
ncov=3

# generate a data set
```

```

set.seed(111) #random seed is reset

dat <- rdemmix(nn,p,ng,distr,mu,sigma)

# the following code can be used to get singular data (remarked off)
# dat[1:300,2]<--4
# dat[300+1:300,1]<-2
## dat[601:1000,1]<-0
## dat[601:1000,2]<-0

#fit the data using KMEANS to get the initial partitions (10 trials)
obj <- EmSkew(dat,ng,distr,ncov,itmax=1000,epsilon=1e-5,nkmeans=10)

# alternatively, if we define initial values like
initobj<-list()

initobj$pro <- pro
initobj$mu <- mu
initobj$sigma<- sigma

initobj$dof <- dof
initobj$delta<- delta

# then we can fit the data from initial values
obj <- EmSkew(dat,ng,distr,ncov,init=initobj,itmax=1000,epsilon=1e-5)

# finally, if we know initial partition such as
clust <- rep(1:ng,nn)

# then we can fit the data from given initial partition
obj <- EmSkew(dat,ng,distr,ncov,clust=clust,itmax=1000,epsilon=1e-5)

# plot the 2D contours

colnames(dat)<- paste("x",1:p,sep='')

## Not run:
x11()
EmSkew.flow(dat,obj)

## End(Not run)

```

## Description

Contour of fitted mixture density can be an important indicator of goodness-of-fit. Together with the heatmaps, this function provides a general idea of how well the mixture model fits to the data.

## Usage

```
EmSkew.flow(S, obj = NULL, distr="", diag.panel=TRUE, upper.panel="type2",
lower.panel = "type3", levels=seq(5, 95, by=20), attop=FALSE, clust=NULL,
title="", path="", plot=TRUE)
EmSkew.contours(S, obj = NULL, clust = NULL, distr="", diag.panel=TRUE,
upper.panel="type2", lower.panel="type3", levels=seq(5, 95, by=20),
plot=TRUE, title="", path='', attop =FALSE)

EmSkew.filter(S, g=1, distr="mst", diag.panel=TRUE, upper.panel="type2",
lower.panel = "type3", levels = 90, attop = FALSE,
title="", path="", plot=TRUE)
```

## Arguments

|                          |   |
|--------------------------|---|
| <code>g</code>           | The number of components to filter the data at FSC and SSC channels.                        |
| <code>S</code>           | A dataframe.  |
| <code>obj</code>         | A list including the parameters of a (skew) mixture model.                                  |
| <code>distr</code>       | A three letter string specifying the distribution.  |
| <code>diag.panel</code>  | A logical value, plot density.  |
| <code>upper.panel</code> | A string for the panel to be used in upper triangle.  |
| <code>lower.panel</code> | A string for the panel to be used in lower triangle.  |
| <code>levels</code>      | A vector of contour percentage levels for the plots. It should be in the range of 0 to 100. |
| <code>attop</code>       | A logical value indicating the direction of diagonal panels.                                |
| <code>clust</code>       | A vector with the cluster labels for each observation of the sample.                        |
| <code>title</code>       | The png file name.  |
| <code>path</code>        | The path to the folder where plots are stored.  |
| <code>plot</code>        | A logical variable, whether plot it in the windows.   |

## Details

R package `geneplotter` is required.

## Note

The input list 'obj' must be assigned a component "distr".

## See Also

[conplot](#)

## Description

The engines to fit the data into mixture models using initial partition or initial values. set.

## Usage

```
EmSkewfit1(dat, g, clust,          distr, ncov, itmax, epsilon, initloop=20)
EmSkewfit2(dat, g,          init, distr, ncov, itmax, epsilon)
```

## Arguments

|                       |  |
|-----------------------|--|
| <code>dat</code>      | The dataset, an $n$ by $p$ numeric matrix, where $n$ is number of observations and $p$ the dimension of data.                                    |
| <code>g</code>        | The number of components of the mixture model  |
| <code>distr</code>    | A three letter string indicating the type of distribution to be fit. See Details.  |
| <code>ncov</code>     | A small integer indicating the type of covariance structure. See Details.  |
| <code>clust</code>    | A vector of integers specifying the initial partitions of the data   |
| <code>init</code>     | A list containing the initial parameters for the mixture model. See details.   |
| <code>itmax</code>    | A big integer specifying the maximum number of iterations to apply   |
| <code>epsilon</code>  | A small number used to stop the EM algorithm loop when the relative difference between log-likelihood at each iteration become sufficient small. |
| <code>initloop</code> | A integer specifying the number of initial loops   |

## Details

The distribution type, determined by the `distr` parameter, which may take any one of the following values: "mvn" for a multivariate normal, "mvt" for a multivariate t-distribution, "msn" for a multivariate skew normal distribution and "mst" for a multivariate skew t-distribution.

The covariance matrix type, represented by the `ncov` parameter, may be any one of the following: `ncov=1` for a common variance, `ncov=2` for a common diagonal variance, `ncov=3` for a general variance, `ncov=4` for a diagonal variance, `ncov=5` for  $\sigma(h) \cdot I(p)$  (diagonal covariance with same identical diagonal element values).

The parameter `init` is a list with elements: `pro`, a numeric vector of the mixing proportion of each component; `mu`, a  $p$  by  $g$  matrix with each column as its corresponding mean; `sigma`, a three dimensional  $p$  by  $p$  by  $g$  array with its  $j$ th component matrix ( $p, p, j$ ) as the covariance matrix for  $j$ th component of mixture models; `dof`, a vector of degrees of freedom for each component; `delta`, a  $p$  by  $g$  matrix with its columns corresponding to skew parameter vectors.

## Value

|                    |   |
|--------------------|---|
| <code>error</code> | Error code, 0 = normal exit; 1 = did not converge within <code>itmax</code> iterations; 2 = failed to get the initial values; 3 = singularity |
| <code>aic</code>   | Akaike Information Criterion (AIC)  |
| <code>bic</code>   | Bayes Information Criterion (BIC)   |

|        |  |
|--------|--|
| pro    | A vector of mixing proportions, see Details.   |
| mu     | A numeric matrix with each column corresponding to the mean, see Details.  |
| sigma  | An array of dimension (p,p,g) with first two dimension corresponding covariance matrix of each component, see Details. |
| dof    | A vector of degrees of freedom for each component, see Details.  |
| delta  | A p by g matrix with each column corresponding to a skew parameter vector, see Details.                                |
| clust  | A vector of final partition  |
| loglik | The loglikelihood at convergence   |
| lk     | A vector of loglikelihood at each EM iteration   |
| tau    | An n by g matrix of posterior probability for each data point  |

## References

- McLachlan G.J. and Krishnan T. (2008). The EM Algorithm and Extensions (2nd). New Jersey: Wiley.
- McLachlan G.J. and Peel D. (2000). Finite Mixture Models. New York: Wiley.

## See Also

`init.mix, initEmmix, EmSkew, rdemmix, rdemmix2, rdmvn, rdmvt, rdmsn, rdmst.`

## Examples

```
n1=300;n2=300;n3=400;
nn <-c(n1,n2,n3)
n=1000
p=2
ng=3

sigma<-array(0,c(2,2,3))
for(h in 2:3) sigma[, ,h]<-diag(2)
sigma[, ,1]<-cbind( c(1,0),c(0,1))
mu <- cbind(c(4,-4),c(3.5,4),c( 0, 0))

# for other distributions,
#delta <- cbind(c(3,3),c(1,5),c(-3,1))
#dof <- c(3,5,5)

pro <- c(0.3,0.3,0.4)

distr="mvn"
ncov=3

#first we generate a data set
set.seed(111) #random seed is set
dat <- rdemmix(nn,p,ng,distr,mu,sigma,dof=NULL,delta=NULL)
```



```

#start from initial partition
clust<- rep(1:ng,nn)
obj1 <- EmSkewfit1(dat, ng, clust, distr, ncov, itmax=1000, epsilon=1e-4)

#start from initial values
#alternatively, if we define initial values like

init<-list()

init$pro<-pro
init$mu<-mu
init$sigma<-sigma

# for other distributions,
#delta <- cbind(c(3,3),c(1,5),c(-3,1))
#dof    <- c(3,5,5)
#init$dof<-dof
#init$delta<-delta

obj2 <-EmSkewfit2(dat, ng, init, distr, ncov,itmax=1000, epsilon=1e-4)

```

---

EmSkewMOD

---

*Calculate modes*


---

## Description

Calculate the mode points for each component of skew mixture models.

## Usage

```
EmSkewMOD(p,g,distr,mu,sigma,dof,delta,nrand=10000)
```

## Arguments

|       |   |
|-------|---|
| p     | The dimension of the data   |
| g     | The number of components to be fit  |
| distr | A three letter string of distribution id  |
| mu    | A numeric matrix with each column corresponding to the mean   |
| sigma | An array of dimension (p,p,g) with first two dimensions corresponding covariance matrix of each component |
| dof   | A vector of degrees of freedom for each component   |
| delta | A matrix with each column as skew parameter vector  |
| nrand | The number of random numbers.   |

## Value

A p by g matrix of mode points for each component of the skew mixture model.

**Examples**

```

p=2
g=3

sigma<-array(0,c(2,2,3))
for(h in 2:3) sigma[,h]<-diag(2)
sigma[,1]<-cbind( c(1,0),c(0,1))
mu  <- cbind(c(4,-4),c(3.5,4),c( 0, 0))

delta <- cbind(c(3,3),c(1,5),c(-3,1))
dof   <- c(3,5,5)

distr="mst"

EmSkewMOD(p,g,distr,mu,sigma,dof,delta,nrand=10000)

```

---

|            |                   |
|------------|-------------------|
| error.rate | <i>Error Rate</i> |
|------------|-------------------|

---

**Description**

Calculate the Error Rate of a partition

**Usage**

```

error.rate(clust1,clust2)
rand.index(LabelA,LabelB)

```

**Arguments**

|        |   |
|--------|---|
| clust1 | An integer vector of cluster label 1            |
| clust2 | An integer vector of cluster label 2            |
| LabelA | An integer vector of the true membership labels |
| LabelB | An integer vector of the predicted labels       |

**Details**

clust1 and clust 2 must match, i.e, same number of clusters

**Value**

error.rate gives Error Rate

**Examples**

```

clu1<-c(1,2,3,1,1,2,2,3,3)
clu2<-c(2,2,2,1,1,1,3,3,3)
error.rate(clu1, clu2)
rand.index(clu1, clu2)

```

getcov

*Covariance***Description**

Recalculate covariance for a given structure.

**Usage**

```
getcov(msigma, sumtau, n, p, g, ncov)
```

**Arguments**

|        |   |
|--------|---|
| msigma | An array for the covariance matrices.                                 |
| sumtau | A vector for the expected number of observations from each component. |
| n      | An integer for the sample size.                                       |
| p      | An integer for the dimension.   |
| g      | An integer for the number of components.                              |
| ncov   | An integer for type of covariance.                                    |

**Value**

The covariance array.

**References**

McLachlan G.J. and Krishnan T. (2008). The EM Algorithm and Extensions (2nd). Hoboken, New Jersey: Wiley.

McLachlan G.J. and Peel D. (2000). Finite Mixture Models. New York: Wiley.

getICL

*The ICL criterion***Description**

Calculate the Integrated Completed Likelihood(ICL) criterion

**Usage**

```
getICL(x, n, p, g, distr, ncov, pro, mu, sigma, dof, delta, clust)
```

**Arguments**

|                    |  |
|--------------------|--|
| <code>x</code>     | An n by p data matrix  |
| <code>n</code>     | The total number of points   |
| <code>p</code>     | Dimension of data  |
| <code>g</code>     | the number of components of the mixture model  |
| <code>distr</code> | A three letter string indicating the type of distribution to be fit.                                     |
| <code>ncov</code>  | A small integer indicating the type of covariance structure.   |
| <code>pro</code>   | A vector of mixing proportions   |
| <code>mu</code>    | A numeric matrix with each column corresponding to the mean  |
| <code>sigma</code> | An array of dimension (p,p,g) with first two dimension corresponding covariance matrix of each component |
| <code>dof</code>   | A vector of degrees of freedom for each component  |
| <code>delta</code> | A p by g matrix with each column corresponding to a skew parameter vector                                |
| <code>clust</code> | A vector of partition  |

**Value**

|     |           |
|-----|-----------|
| ICL | ICL value |
|-----|-----------|

**References**

Biernacki C. Celeux G., and Govaert G. (2000). Assessing a Mixture Model for Clustering with the integrated Completed Likelihood. IEEE Transactions on Pattern Analysis and Machine Intelligence. 22(7). 719-725.

**Examples**

```
n1=300;n2=300;n3=400;
nn <-c(n1,n2,n3)
n=1000
p=2
ng=3

sigma<-array(0,c(2,2,3))
for(h in 2:3) sigma[, ,h]<-diag(2)
sigma[, ,1]<-cbind( c(1,0),c(0,1))
mu <- cbind(c(4,-4),c(3.5,4),c( 0, 0))

pro <- c(0.3,0.3,0.4)

distr="mvn"
ncov=3

#first we generate a data set
set.seed(111) #random seed is set
dat <- rdemmix(nn,p,ng,distr,mu,sigma,dof=NULL,delta=NULL)

#start from initial partition
clust<- rep(1:ng,nn)
obj <- EmSkewfit1(dat, ng, clust, distr, ncov, itmax=1000,epsilon=1e-4)
```

```
getICL(dat,n,p,ng, distr,ncov,obj$pro,obj$mu,obj$sigma,obj$dof,
obj$delta,obj$clust)
```

getSWR

*Calculate modes***Description**

Calculate the distances.

**Usage**

```
getSWR(dat,g,sigma,clust, tau)
intradist(dat,g,sigma, clust, tau)
interdist(dat,g,sigma, clust, tau)
mahalonobis(p, g, mu, sigma)
```

**Arguments**

|       |   |
|-------|---|
| dat   | The dataset, an n by p numeric matrix, where n is number of observations and p the dimension of data.     |
| p     | The dimension of the data   |
| g     | The number of components to be fit  |
| mu    | A numeric matrix with each column corresponding to the mean   |
| sigma | An array of dimension (p,p,g) with first two dimensions corresponding covariance matrix of each component |
| clust | A vector of integers specifying the initial partitions of the data; the default is NULL.                  |
| tau   | An n by g matrix of posterior probability for each data point   |

initEmmix

*Initialize Emmix Parameters***Description**

Obtains initial parameter set for use in the EM algorithm. Grouping of the data occurs through one of three possible clustering methods: k-means, random start, and hierarchical clustering.

**Usage**

```
initEmmix(dat, g, clust, distr, ncov,maxloop=20)
init.mix( dat, g, distr, ncov, nkmeans, nrandom, nhclust,maxloop=20)
```

## Arguments

|                      |   |
|----------------------|---|
| <code>dat</code>     | The dataset, an $n$ by $p$ numeric matrix, where $n$ is number of observations and $p$ the dimension of data.                     |
| <code>g</code>       | The number of components of the mixture model   |
| <code>distr</code>   | A three letter string indicating the type of distribution to be fit. See Details.   |
| <code>ncov</code>    | A small integer indicating the type of covariance structure. See Details.   |
| <code>clust</code>   | An initial partition of the data  |
| <code>nkmeans</code> | An integer to specify the number of KMEANS partitions to be used to find the best initial values                                  |
| <code>nrandom</code> | An integer to specify the number of random partitions to be used to find the best initial values                                  |
| <code>nhclust</code> | A logical value to specify whether or not to use hierarchical cluster methods. If TRUE, the Complete Linkage method will be used. |
| <code>maxloop</code> | An integer to specify how many iterations to be tried to find the initial values, the default value is 10.                        |

## Details

The distribution type, determined by the `distr` parameter, which may take any one of the following values: "mvn" for a multivariate normal, "mvt" for a multivariate t-distribution, "msn" for a multivariate skew normal distribution and "mst" for a multivariate skew t-distribution.

The covariance matrix type, represented by the `ncov` parameter, may be any one of the following: `ncov=1` for a common variance, `ncov=2` for a common diagonal variance, `ncov=3` for a general variance, `ncov=4` for a diagonal variance, `ncov=5` for  $\sigma(h)*I(p)$  (diagonal covariance with same identical diagonal element values).

The return values include following components: `pro`, a numeric vector of the mixing proportion of each component; `mu`, a  $p$  by  $g$  matrix with each column as its corresponding mean; `sigma`, a three dimensional  $p$  by  $p$  by  $g$  array with its  $j$ th component matrix ( $p, p, j$ ) as the covariance matrix for  $j$ th component of mixture models; `dof`, a vector of degrees of freedom for each component; `delta`, a  $p$  by  $g$  matrix with its columns corresponding to skew parameter vectors.

When the dataset is huge, it becomes time-consuming to use a large `maxloop` to try every initial partition. The default is 10. During the procedure to find the best initial clustering and initial values, for t-distribution and skew t-distribution, we don't estimate the degrees of freedom `dof`, instead they are fixed at 4 for each component.

## Value

|                    |  |
|--------------------|--|
| <code>pro</code>   | A vector of mixing proportions, see Details.   |
| <code>mu</code>    | A numeric matrix with each column corresponding to the mean, see Details.  |
| <code>sigma</code> | An array of dimension $(p, p, g)$ with first two dimension corresponding covariance matrix of each component, see Details. |
| <code>dof</code>   | A vector of degrees of freedom for each component, see Details.  |
| <code>delta</code> | A $p$ by $g$ matrix with each column corresponding to a skew parameter vector, see Details.                                |

## References

McLachlan G.J. and Krishnan T. (2008). The EM Algorithm and Extensions (2nd). New Jersey: Wiley.

McLachlan G.J. and Peel D. (2000). Finite Mixture Models. New York: Wiley.

## See Also

[EmSkew](#)

## Examples

```
sigma<-array(0,c(2,2,3))
for(h in 2:3) sigma[,h]<-diag(2)
sigma[,1]<-cbind(c(1,0.2),c(0.2,1))
mu <- cbind(c(4,-4),c(3.5,4),c(0,0))
delta <- cbind(c(3,3),c(1,5),c(-3,1))
dof <- c(3,5,5)
pro <- c(0.3,0.3,0.4)
n1=300;n2=300;n3=400;
nn<-c(n1,n2,n3)
n=1000
p=2
ng=3
distr="mvn"
ncov=3
#first we generate a data set
set.seed(111) #random seed is set
dat <- rdemmix(nn,p,ng,distr,mu,sigma,dof,delta)
clust<- rep(1:ng,nn)
initobj1 <- initEmmix(dat,ng,clust,distr, ncov)
initobj2 <- init.mix( dat,ng,distr,ncov,nkmeans=10,nrandom=0,nhclust=FALSE)
```

---

inverse

*Inverse of a covariance matrix*

---

## Description

Calculate the inverse of a covariance matrix.

## Usage

```
inverse(sigma, p)
```

## Arguments

|       |                              |
|-------|------------------------------|
| sigma | The covariance matrix.       |
| p     | The dimension of the matrix. |

## Value

The inverse of the covariance matrix.

**Note**

The covariance matrix may be singular. This is of use only for the clustering of the data.

**Author(s)**

Kui Wang

**Examples**

```
a<- matrix(c(1,0,0,0),ncol=2)
a
inverse(a,2)
```

---

|         |                           |
|---------|---------------------------|
| mvt.dof | <i>Degrees of freedom</i> |
|---------|---------------------------|

---

**Description**

Calculate the degrees of freedom

**Usage**

```
mvt.dof(sumtau, sumlnv, lx = 2+1e-04, ux = 200)
```

**Arguments**

|        |              |
|--------|--------------|
| sumtau | A quantity.  |
| sumlnv | A quantity.  |
| lx     | Lower limit. |
| ux     | Upper limit. |

**Details**

It is called by msmvt, jcamvt and jcamst. When there is no solution between lx and ux, ux will be returned.

**Value**

degrees of freedom.

**Note**

It is called by msmvt, jcamst and jcamvt.



rdemmix

*Simulate Data Using Mixture Models***Description**

Generate random number from specified mixture models, including univariate and multivariate Normal distribution, t-distribution, Skew Normal distribution, and Skew t-distribution.

**Usage**

```
rdemmix(nvect, p, g, distr, mu, sigma, dof=NULL, delta=NULL)
rdemmix2(n, p, g, distr, pro, mu, sigma, dof=NULL, delta=NULL)
rdemmix3(n, p, g, distr, pro, mu, sigma, dof=NULL, delta=NULL)
```

**Arguments**

|                    |  |
|--------------------|--|
| <code>nvect</code> | A vector of how many points in each cluster, $c(n_1, n_2, \dots, n_g)$   |
| <code>n</code>     | The total number of points   |
| <code>p</code>     | Dimension of data  |
| <code>g</code>     | The number of clusters   |
| <code>distr</code> | A three letter string indicating the distribution type   |
| <code>pro</code>   | A vector of mixing proportions, see Details.   |
| <code>mu</code>    | A numeric matrix with each column corresponding to the mean, see Details.  |
| <code>sigma</code> | An array of dimension $(p, p, g)$ with first two dimension corresponding covariance matrix of each component, see Details. |
| <code>dof</code>   | A vector of degrees of freedom for each component, see Details.  |
| <code>delta</code> | A $p$ by $g$ matrix with each column corresponding to a skew parameter vector, see Details.                                |

**Details**

The distribution type, determined by the `distr` parameter, which may take any one of the following values: "mvn" for a multivariate normal, "mvt" for a multivariate t-distribution, "msn" for a multivariate skew normal distribution and "mst" for a multivariate skew t-distribution. `pro`, a numeric vector of the mixing proportion of each component; `mu`, a  $p$  by  $g$  matrix with each column as its corresponding mean; `sigma`, a three dimensional  $p$  by  $p$  by  $g$  array with its  $j$ th component matrix  $(p, p, j)$  as the covariance matrix for  $j$ th component of mixture models; `dof`, a vector of degrees of freedom for each component; `delta`, a  $p$  by  $g$  matrix with its columns corresponding to skew parameter vectors.

**Value**

both `rdemmix` and `rdemmix2` return an  $n$  by  $p$  numeric matrix of generated data;  
`rdemmix3` gives a list with components `data`, the generated data, and `cluster`, the clustering of data.

## References

- McLachlan G.J. and Krishnan T. (2008). The EM Algorithm and Extensions (2nd). New Jersey: Wiley.
- McLachlan G.J. and Peel D. (2000). Finite Mixture Models. New York: Wiley.

## See Also

[rdmvn](#), [rdmvt](#), [rdmsn](#), [rdmst](#).

## Examples

```
#specify the dimension of data, and number of clusters
#the number of observations in each cluster
n1=300;n2=300;n3=400;
nn<-c(n1,n2,n3)

p=2
g=3

#specify the distribution
distr <- "mvn"

#specify mean and covariance matrix for each component

sigma<-array(0,c(2,2,3))
for(h in 2:3) sigma[, ,h]<-diag(2)
sigma[, ,1]<-cbind( c(1,-0.1),c(-0.1,1))
mu <- cbind(c(4,-4),c(3.5,4),c( 0, 0))

#reset the random seed
set.seed(111)
#generate the dataset
dat <- rdemmix(nn,p,g,distr, mu,sigma)

# alternatively one can use
pro <- c(0.3,0.3,0.4)
n=1000
set.seed(111)
dat <- rdemmix2(n,p,g,distr,pro,mu,sigma)
plot(dat)

# and

set.seed(111)
dobj <- rdemmix3(n,p,g,distr,pro,mu,sigma)
plot(dobj$data)

#other distributions such as "mvt", "msn", and "mst".

#t-distributions
```

```
dof      <- c(3,5,5)
dat <- rdemmix2(n,p,g,"mvt",pro,mu,sigma,dof)
plot(dat)

#Skew Normal distribution
delta <- cbind(c(3,3),c(1,5),c(-3,1))
dat <- rdemmix2(n,p,g,"msn",pro,mu,sigma,delta=delta)
plot(dat)

#Skew t-distribution
dat <- rdemmix2(n,p,g,"mst",pro,mu,sigma,dof,delta)
plot(dat)
```

# Index

## \*Topic cluster

- bootstrap, 2
- ddmix, 5
- ddmsn, 6
- ddmst, 7
- ddmvn, 8
- ddmvt, 9
- EmSkew, 10
- EmSkewfit, 15
- EmSkewMOD, 17
- error.rate, 18
- getICL, 19
- getSWR, 21
- initEmmix, 21
- rdemmix, 25

## \*Topic datasets

- bootstrap, 2
- ddmsn, 6
- ddmst, 7
- ddmvn, 8
- ddmvt, 9
- EmSkew, 10
- EmSkewfit, 15
- EmSkewMOD, 17
- error.rate, 18
- getICL, 19
- getSWR, 21
- initEmmix, 21
- rdemmix, 25

bootstrap, 2

conplot, 4, 14

conplot2 (conplot), 4

conplot3 (conplot), 4

ddmix, 5

ddmsn, 6, 6, 8–10

ddmst, 6, 7, 7, 9, 10

ddmvn, 6, 7, 8, 8, 10

ddmvt, 6–8, 9, 9

EmSkew, 3, 10, 16, 23

EmSkew.contours, 13

EmSkew.filter (EmSkew.contours),  
13

EmSkew.flow, 5

EmSkew.flow (EmSkew.contours), 13

EmSkewfit, 15

EmSkewfit1 (EmSkewfit), 15

EmSkewfit2 (EmSkewfit), 15

EmSkewMOD, 17

error.rate, 18

getcov, 19

getICL, 19

getSWR, 21

init.mix, 16

init.mix (initEmmix), 21

initEmmix, 12, 16, 21

interdist (getSWR), 21

intradist (getSWR), 21

inverse, 23

mahalonobis (getSWR), 21

mvt.dof, 24

mypanel2 (conplot), 4

mypanel3 (conplot), 4

mypanel4 (conplot), 4

panel.density (conplot), 4

rand.index (error.rate), 18

rdemmix, 3, 7–10, 12, 16, 25

rdemmix2, 16

rdemmix2 (rdemmix), 25

rdemmix3 (rdemmix), 25

rdmsn, 8–10, 16, 26

rdmsn (ddmsn), 6

rdmst, 7, 9, 10, 16, 26

rdmst (ddmst), 7

rdmvn, 7, 8, 10, 16, 26

rdmvn (ddmvn), 8

rdmvt, 7–9, 16, 26

rdmvt (ddmvt), 9