

# Package ‘GMCM’

January 27, 2015

**Type** Package

**Title** Fast estimation of Gaussian Mixture Copula Models

**Description** Unsupervised Clustering and Meta-analysis using Gaussian Mixture Copula Models.

**Version** 1.1.1

**Date** 2014-08-16

**Author** Anders Ellern Bilgrau, Martin Boegsted, Poul Svante Eriksen

**Maintainer** Anders Ellern Bilgrau <abilgrau@math.aau.dk>

**URL** <https://github.com/AEBilgrau/GMCM>,  
<http://people.math.aau.dk/~abilgrau>

**BugReports** <https://github.com/AEBilgrau/GMCM/issues>

**License** GPL (>= 2)

**Imports** Rcpp (>= 0.10.6)

**LinkingTo** Rcpp, RcppArmadillo

**Suggests** testthat (>= 0.3)

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2014-10-06 17:42:51

## R topics documented:

GMCM-package	2
choose.theta	3
dmvnormal	5
EMAlgorithm	6
fit.full.GMCM	7
fit.meta.GMCM	9
freshVsFrozen	11
full2meta	12
get.IDR	13

is.theta . . . . .	15
rtheta . . . . .	16
SimulateGMCMData . . . . .	17
u133VsExon . . . . .	19
Uhat . . . . .	20
<b>Index</b>	<b>21</b>

GMCM-package

*Fast optimization of Gaussian Mixture Copula Models*

## Description

Gaussian mixture copula models (GMCM) can be used for unsupervised clustering and meta analysis. In meta analysis, GMCMs are used to identify features which have been reproduced across multiple experiments. This package provides a fast and general implementation of GMCM cluster analysis and serves as an extension of the features available in the `idr` package.

## Details

If the meta analysis of Li et al. (2011) is to be performed, the function `fit.meta.GMCM` is used to identify the maximum likelihood estimate of the special Gaussian mixture copula model (GMCM) defined by Li et al. (2011). The function `get.IDR` computes the local and adjusted Irreproducible Discovery Rates defined by Li et al. (2011) to determine the level of reproducibility.

Tewari et. al. (2011) proposed using GMCMs as a general unsupervised clustering tool. If such a general unsupervised clustering is needed, like above, the function `fit.full.GMCM` computes the maximum likelihood estimate of the general GMCM. The function `get.prob` is used to estimate the class membership probabilities of each observation.

`SimulateGMCMData` provide easy simulation from the GMCMs.

## Author(s)

Anders Ellern Bilgrau, Martin Boegsted, Poul Svante Eriksen

Maintainer: Anders Ellern Bilgrau <abilgrau@math.aau.dk>

## References

Li, Q., Brown, J. B. J. B., Huang, H., & Bickel, P. J. (2011). Measuring reproducibility of high-throughput experiments. *The Annals of Applied Statistics*, 5(3), 1752-1779. doi:10.1214/11-AOAS466

Tewari, A., Giering, M. J., & Raghunathan, A. (2011). Parametric Characterization of Multimodal Distributions with Non-gaussian Modes. 2011 IEEE 11th International Conference on Data Mining Workshops, 286-292. doi:10.1109/ICDMW.2011.135

**See Also**

Core user functions: [fit.meta.GMCM](#), [fit.full.GMCM](#), [get.IDR](#), [get.prob](#), [SimulateGMCMData](#), [SimulateGMMDData](#), [rtheta](#), [Uhat](#), [choose.theta](#), [full2meta](#), [meta2full](#)

Package by Li et. al. (2011): [idr](#).

**Examples**

```
# Loading data
data(u133VsExon)

# Subsetting data to reduce computation time
u133VsExon <- u133VsExon[1:5000, ]

# Ranking and scaling,
# Remember large values should be critical to the null!
uhat <- Uhat(1 - u133VsExon)

# Visualizing P-values and the ranked and scaled P-values
## Not run:
par(mfrow = c(1,2))
plot(u133VsExon, cex = 0.5, pch = 4, col = "tomato", main = "P-values"
     xlab = "P (U133)", ylab = "P (Exon)")
plot(uhat, cex = 0.5, pch = 4, col = "tomato", main = "Ranked P-values",
     xlab = "rank(1-P) (U133)", ylab = "rank(1-P) (Exon)")

## End(Not run)

# Fitting using BFGS
fit <- fit.meta.GMCM(uhat, init.par = c(0.5, 1, 1, 0.5), pgtol = 1e-2,
                    method = "L-BFGS", positive.rho = TRUE, verbose = TRUE)

# Compute IDR values and classify
idr <- get.IDR(uhat, par = fit)
table(idr$K) # 1 = irreproducible, 2 = reproducible

## Not run:
# See clustering results
par(mfrow = c(1,2))
plot(u133VsExon, cex = 0.5, pch = 4, main = "Classified genes",
     col = c("tomato", "steelblue")[idr$K],
     xlab = "P (U133)", ylab = "P (Exon)")
plot(uhat, cex = 0.5, pch = 4, main = "Classified genes",
     col = c("tomato", "steelblue")[idr$K],
     xlab = "rank(1-P) (U133)", ylab = "rank(1-P) (Exon)")

## End(Not run)
```

**Description**

This function uses a k-means algorithm to heuristically select suitable starting values for the general model.

**Usage**

```
choose.theta(u, m, ...)
```

**Arguments**

u	A matrix of (estimates of) realizations from the GMCMM.
m	The number of components to be fitted.
...	Arguments passed to <a href="#">kmeans</a> .

**Details**

The function selects the centers from the k-means algorithm as an initial estimate of the means. The proportional sizes of the clusters are selected as the initial values of the mixture proportions. The within cluster standard deviations are used as the variance of the clusters. The correlations between each dimension are taken to be zero.

**Value**

A list of parameters for the GMCMM model on the form described in [rtheta](#).

**Note**

The function uses the `kmeans` function from the `stats`-package.

**Author(s)**

Anders Ellern Bilgrau ([abilgrau@math.aau.dk](mailto:abilgrau@math.aau.dk))

**Examples**

```
set.seed(2)

# Simulating data
data1 <- SimulateGMCMDData(n = 10000, m = 3, d = 2)
obs.data <- Uhat(data1$u) # The ranked observed data

# Using choose.theta to get starting estimates
theta <- choose.theta(u = obs.data, m = 3)
print(theta)

# To illustrate theta, we simulate from the model
data2 <- SimulateGMMData(n = 10000, theta = theta)

cols <- apply(get.prob(obs.data, theta), 1, which.max)
```

```
# Plotting
par(mfrow = c(1,3))
plot(data1$z, main = "True latent GMM")
plot(Uhat(data1$u), col = cols,
     main = "Observed GMCM\nColoured by k-means clustering")
plot(data2$z, main = "initial GMM")
```

---

dmvnormal

*Multivariate Gaussian probability density function and simulation*

---

### Description

Fast simulation and evaluation of multivariate Gaussian probability densities.

### Usage

```
dmvnormal(x, mu, sigma)
```

```
rmvnormal(n, mu, sigma)
```

### Arguments

x	A p times k matrix of quantiles. Each rows correspond to a realization from the density and each column corresponds to a dimension.
mu	The mean vector of dimension k.
sigma	The variance-covariance matrix of dimension k times k.
n	The number of observations to be simulated.

### Details

dmvnormal functions similarly to dmvnorm from the mvtnorm-package and likewise for rmvnormal and rmvnorm.

### Value

dmvnormal returns a 1 by p matrix of the probability densities corresponding to each row of x. sigma. Each row corresponds to an observation.

rmvnormal returns a p by k matrix of observations from a multivariate normal distribution with the given mean mu and covariance

### Author(s)

Anders Ellern Bilgrau

### See Also

dmvnorm and rmvnorm in the mvtnorm-package.

**Examples**

```
dmvnormal(x = matrix(rnorm(300), 100, 3),
           mu = 1:3,
           sigma = diag(3))
rmvnormal(n = 10, mu = 1:4, sigma = diag(4))
```

EMAlgorithm

*EM algorithm for Gaussian mixture models***Description**

The regular expectation-maximization algorithm for general multivariate Gaussian mixture models.

**Usage**

```
EMAlgorithm(x, theta, eps = 1e-06, max.ite = 1e+05, trace.theta = FALSE,
           verbose = FALSE)
```

**Arguments**

x	A matrix of observations where rows correspond to features and columns to experiments.
theta	A list of parameters as described in <a href="#">rtheta</a> .
eps	The maximal required difference in successive likelihoods to establish convergence.
max.ite	The maximum number of iterations.
trace.theta	Logical. If TRUE, all estimates are stored and returned. Default is FALSE.
verbose	Set to TRUE for verbose output. Default is FALSE.

**Details**

Though not as versatile, the algorithm can be a faster alternative to Mclust in the mclust-package.

**Value**

A list of length 3 with elements:

theta	A list of the estimated parameters as described in <a href="#">rtheta</a> .
loglik.tr	A numeric vector of the log-likelihood trace.
kappa	A matrix where $\kappa[i, j]$ is the probability that $x[i, ]$ is realized from the $j$ 'th component.

**Author(s)**

Anders Ellern Bilgrau (abilgrau@math.aau.dk)

**See Also**

[rtheta](#), [PseudoEMAlgorithm](#)

**Examples**

```
set.seed(10)
data <- SimulateGMCMData(n = 1000, d = 2, m = 3)
start.theta <- rtheta(d = 2, m = 3)
res <- GMCM::EMAlgorithm(data$z, theta = start.theta)

par(mfrow = c(1,2))
plot(data$z, cex = 0.5, pch = 16, main = "Simulated data",
      col = rainbow(3)[data$K])
plot(data$z, cex = 0.5, pch = 16, main = "GMM clustering",
      col = rainbow(3)[apply(res$kappa,1,which.max)])
```

---

fit.full.GMCM

*Unsupervised clustering using a general GMCM*


---

**Description**

Perform unsupervised clustering using various optimization procedures to find the maximum likelihood estimate of the general Gaussian mixture copula model by Tewari et al. (2011).

**Usage**

```
fit.full.GMCM(u, m, theta = choose.theta(u, m), method = c("NM", "SANN",
  "L-BFGS", "L-BFGS-B", "PEM"), max.ite = 1000, verbose = TRUE, ...)
```

**Arguments**

u	An n by d matrix of ranked and scaled test statistics. Rows correspond to observations and columns to the dimensions of the variables.
m	The number of components to be fitted.
theta	A list of parameters as defined in <a href="#">rtheta</a> . If theta is not provided, then heuristic starting values are chosen using the k-means algorithm.
method	A character vector of length 1. The optimization method used. Should be either "NM", "SANN", "L-BFGS", "L-BFGS-B", or "PEM" which are the Nelder-Mead, Simulated Annealing, limited-memory quasi-Newton method, limited-memory quasi-Newton method with box constraints, and the pseudo EM algorithm, respectively. Default is "NM". See <a href="#">optim</a> for further details.
max.ite	The maximum number of iterations. If the method is "SANN" this is the number of iterations as there is no other stopping criterion. (See <a href="#">optim</a> )
verbose	Logical. If TRUE, a trace of the parameter estimates is made.
...	Arguments passed to the control-list in <a href="#">optim</a> or <a href="#">PseudoEMAlgorithm</a> if the method is "PEM".

**Details**

The "L-BFGS-B" method does not perform a transformation of the parameters and uses box-constraints as implemented in `optim`.

Note that the many parameter configurations are poorly estimable or directly unidentifiable.

**Value**

A list of parameters formatted as described in `rtheta`.

**Note**

All the optimization procedures are strongly dependent on the initial values and the cooling scheme. Therefore it is advisable to apply multiple different initial parameters and select the best fit.

The `choose.theta` itself chooses random initializations. Hence, the output when `theta` is not directly supplied can vary.

See `optim` for further details.

**Author(s)**

Anders Ellern Bilgrau (abilgrau@math.aau.dk)

**References**

Li, Q., Brown, J. B. J. B., Huang, H., & Bickel, P. J. (2011). Measuring reproducibility of high-throughput experiments. *The Annals of Applied Statistics*, 5(3), 1752-1779. doi:10.1214/11-AOAS466

Tewari, A., Giering, M. J., & Raghunathan, A. (2011). Parametric Characterization of Multimodal Distributions with Non-gaussian Modes. 2011 IEEE 11th International Conference on Data Mining Workshops, 286-292. doi:10.1109/ICDMW.2011.135

**See Also**

`optim`, `get.prob`

**Examples**

```
set.seed(17)
sim <- SimulateGMCMData(n = 1000, m = 3, d = 2)

# Plotting simulated data
par(mfrow = c(1,2))
plot(sim$z, col = rainbow(3)[sim$K], main = "Latent process")
plot(sim$u, col = rainbow(3)[sim$K], main = "GMCM process")

# Observed data
uhat <- Uhat(sim$u)

# The model should be fitted multiple times using different starting estimates
start.theta <- choose.theta(uhat, m = 3) # Random starting estimate
```



```

res <- fit.full.GMCM(u = uhat, theta = start.theta,
                    method = "NM", max.ite = 3000,
                    reltol = 1e-2, trace = TRUE) # Note, 1e-2 is too big

# Confusion matrix
Khat <- apply(get.prob(uhat, theta = res), 1, which.max)
table("Khat" = Khat, "K" = sim$K) # Note, some components have been swapped

# Simulation from GMCM with the fitted parameters
simfit <- SimulateGMCMData(n = 1000, theta = res)

# As seen, the underlying latent process is hard to estimate.
# The clustering, however, is very good.
par(mfrow = c(2,2))
plot(simfit$z, col = simfit$K, main = "Model check 1\nSimulated GMM")
plot(simfit$u, col = simfit$K, main = "Model check 2\nSimulated GMCM")
plot(sim$u, col = Khat, main = "MAP clustering")

```

---

fit.meta.GMCM

*Reproducibility/meta analysis using GMCMs*


---

## Description

This function performs reproducibility (or meta) analysis using GMCM. It features various optimization routines to identify the maximum likelihood estimate of the special Gaussian mixture copula model proposed by Li et. al. (2011).

## Usage

```

fit.meta.GMCM(u, init.par, method = c("NM", "SANN", "L-BFGS", "L-BFGS-B",
    "PEM"), max.ite = 1000, verbose = TRUE, positive.rho = TRUE,
    trace.theta = FALSE, ...)

```

## Arguments

<code>u</code>	An $n$ by $d$ matrix of test statistics. Rows correspond to features and columns to experiments. Large values are assumed to be indicative of reproducible genes.
<code>init.par</code>	A 4-dimensional vector of the initial parameters where, <code>init.par[1]</code> is the mixture proportion of spurious signals, <code>init.par[2]</code> is the mean, <code>init.par[3]</code> is the standard deviation, <code>init.par[4]</code> is the correlation.
<code>method</code>	A character vector of length 1. The optimization method used. Should be either "NM", "SANN", "L-BFGS", "L-BFGS-B", or "PEM" which are abbreviations of Nelder-Mead, Simulated Annealing, limited-memory quasi-Newton method, limited-memory quasi-Newton method with box constraints, and the pseudo EM algorithm, respectively. Default is "NM". See <a href="#">optim</a> for further details.
<code>max.ite</code>	The maximum number of iterations. If the method is "SANN" this is the number of iterations as there is no other stopping criterion. (See <a href="#">optim</a> )

verbose	Logical. If TRUE, the log likelihood values are printed.
positive.rho	Logical. If TRUE, the correlation parameter is restricted to be positive.
trace.theta	Logical. Extra convergence information is appended as a list to the output returned if TRUE. The exact behavior is dependent on the value of method. If method equals "PEM", the argument is passed to trace.theta in <a href="#">PseudoEMAlgorithm</a> . Otherwise it is passed to the control argument trace in <a href="#">optim</a> .
...	Arguments passed to the control-list in <a href="#">optim</a> or <a href="#">PseudoEMAlgorithm</a> if method is "PEM".

### Details

The "L-BFGS-B" method does not perform a transformation of the parameters.

### Value

A vector par of length 4 of the fitted parameters where par[1] is the probability of being from the first (or null) component, par[2] is the mean, par[3] is the standard deviation, and par[4] is the correlation.

### Note

Simulated annealing is strongly dependent on the initial values and the cooling scheme. See [optim](#) for further details.

### Author(s)

Anders Ellern Bilgrau (abilgrau@math.aau.dk)

### References

Li, Q., Brown, J. B. J. B., Huang, H., & Bickel, P. J. (2011). Measuring reproducibility of high-throughput experiments. *The Annals of Applied Statistics*, 5(3), 1752-1779. doi:10.1214/11-AOAS466

### See Also

[optim](#)

### Examples

```
set.seed(1)

# True parameters
true.par <- c(0.9, 2, 0.7, 0.6)
# Simulation of data from the GMCM model
data <- SimulateGMCMData(n = 1000, par = true.par)
uhat <- Uhat(data$u) # Ranked observed data

init.par <- c(0.5, 1, 0.5, 0.9) # Initial parameters
```

```

# Optimization with Nelder-Mead
nm.par <- fit.meta.GMCM(uhat, init.par = init.par, method = "NM")

## Not run:
# Comparison with other optimization methods
# Optimization with simulated annealing
sann.par <- fit.meta.GMCM(uhat, init.par = init.par, method = "SANN",
                          max.ite = 3000, temp = 1)
# Optimization with the Pseudo EM algorithm
pem.par <- fit.meta.GMCM(uhat, init.par = init.par, method = "PEM")

# The estimates agree nicely
rbind("True" = true.par, "Start" = init.par,
      "NM" = nm.par, "SANN" = sann.par, "PEM" = pem.par)

## End(Not run)

# Get estimated cluster
Khat <- get.IDR(x = uhat, par = nm.par)$Khat
plot(uhat, col = Khat, main = "Clustering\nIDR < 0.05")

```

---

freshVsFrozen

*Reproducibility between Fresh and Frozen B-cell subtypes*


---

## Description

This dataset contains a data frame of  $t$ -scores (from a Linear mixed effects model) and  $p$ -values for differential expression between pre (Im, N) and post germinal (M, PB) centre cells within peripheral blood. The first and second column contain the the test for the hypothesis of no differentially expression between pre and post germinal cells for the freshly sorted and gene profiled cells. The third and fourth column contain the the test for the hypothesis of no differentially expression between pre and post germinal cells for the cryopreserved (frozen), thawed, sorted, and gene profiled cells. The fifth and sixth column contain the the test for the hypothesis of no differentially expression between fresh and frozen cells. The used array type was Affymetrix Human Exon 1.0 ST microarray.

## Format

The format of the data frame is:

```

'data.frame': 18708 obs. of 6 variables:
 $ PreVsPost.Fresh.tstat : num -1.073 -0.381 -1.105 -0.559 -1.054 ...
 $ PreVsPost.Fresh.pval : num 0.283 0.703 0.269 0.576 0.292 ...
 $ PreVsPost.Frozen.tstat: num -0.245 -0.731 -0.828 -0.568 -1.083 ...
 $ PreVsPost.Frozen.pval: num 0.806 0.465 0.408 0.57 0.279 ...
 $ FreshVsFrozen.tstat : num 0.836 1.135 -0.221 0.191 -0.783 ...
 $ FreshVsFrozen.pval : num 0.403 0.256 0.825 0.849 0.434 ...

```

## Details

Further details can be found in Rasmussen and Bilgrau et al. (2014).

## Author(s)

Anders Ellern Bilgrau <abilgrau (at) math.aau.dk>

## References

Rasmussen SM, Bilgrau AE, Schmitz A, Falgreen S, Bergkvist KS, Tramm AM, Baech J, Jacobsen CL, Gaihede M, Kjeldsen MK, Boedker JS, Dybkaer K, Boegsted M, Johnsen HE (2014). "Stable Phenotype Of B-Cell Subsets Following Cryopreservation and Thawing of Normal Human Lymphocytes Stored in a Tissue Biobank." *Cytometry Part B: Clinical Cytometry*. ISSN 1552-4957. doi:10.1002/cytob.21192. URL <http://dx.doi.org/10.1002/cytob.21192>.

## Examples

```
data(freshVsFrozen)
str(freshVsFrozen)

# Plot P-values
plot(freshVsFrozen[,c(2,4)], cex = 0.5)

# Plot ranked and scaled P-values
plot(Uhat(abs(freshVsFrozen[,c(1,3)])), cex = 0.5)
```

---

full2meta

*Convert between parameter formats*

---

## Description

These functions converts the parameters between the general Gaussian mixture (copula) model and the special GMCM. Most functions of the GMCM packages use the theta format described in [rtheta](#).

## Usage

```
full2meta(theta)
```

```
meta2full(par, d)
```

## Arguments

theta	A list of parameters for the full model. Formatted as described in <a href="#">rtheta</a> .
par	A vector of length 4 where par[1] is the probability of coming from the first component, par[2] is the mean value, par[3] is the standard deviation, and par[4] is the correlation of the reproducible component.
d	The dimension of the mixture distribution.

**Details**

If a `theta` is supplied which is not on the form of Li et. al. (2011) the output is coerced by simply picking the first elements of the first mean vector and first covariance matrix as mean and standard deviation, respectively.

**Value**

`full2meta` returns a numeric vector of length 4 formatted as `par`.

`meta2full` returns a formatted list of parameters as described by [rtheta](#).

**Author(s)**

Anders Ellern Bilgrau ([abilgrau@math.aau.dk](mailto:abilgrau@math.aau.dk))

**References**

Li, Q., Brown, J. B. J. B., Huang, H., & Bickel, P. J. (2011). Measuring reproducibility of high-throughput experiments. *The Annals of Applied Statistics*, 5(3), 1752-1779. doi:10.1214/11-AOAS466

Tewari, A., Giering, M., & Raghunathan, A. (2011). Parametric Characterization of Multimodal Distributions with Non-gaussian Modes. *IEEE 11th International Conference on Data Mining Workshops*, 2011, 286-292. doi:10.1109/ICDMW.2011.135

**See Also**

[rtheta](#)

**Examples**

```
theta <- GMCM:::rtheta(m = 2, d = 2)
print(par <- full2meta(theta))
print(theta.special.case <- meta2full(par, d = 2))
```

---

get.IDR

*Posterior class probabilities and local, and adjusted IDR.*

---

**Description**

Functions for computing posterior cluster probabilities (`get.prob`) in the general GMCM as well as local and adjusted irreproducibility discovery rates (`get.IDR`) in the special GMCM.

**Usage**

```
get.IDR(x, par, threshold = 0.05, ...)
```

```
get.idr(x, theta, ...)
```

```
get.prob(x, theta, ...)
```

**Arguments**

<code>x</code>	A matrix of observations where rows corresponds to features and colums to studies.
<code>par</code>	A vector of length 4 where <code>par[1]</code> is mixture proportion of the irreproducible component, <code>par[2]</code> is the mean value, <code>par[3]</code> is the standard deviation, and <code>par[4]</code> is the correlation of the reproducible component.
<code>threshold</code>	The thresholding level of the IDR rate.
<code>theta</code>	A list of parameters for the full model as described in <a href="#">rtheta</a> .
<code>...</code>	Arguments passed to <a href="#">qgmm.marginal</a> .

**Value**

`get.IDR` returns a list of length 5 with elements:

<code>idr</code>	A vector of the local idr values. I.e. the posterior probability that <code>x[i, ]</code> belongs to the irreproducible component.
<code>IDR</code>	A vector of the adjusted IDR values.
<code>l</code>	The number of reproducible features at the specified <code>threshold</code> .
<code>threshold</code>	The IDR threshold at which features are deemed reproducible.
<code>Khat</code>	A vector signifying whether the corresponding feature is reproducible or not.

`get.prob` returns a matrix where entry  $(i, j)$  is the posterior probability that the observation `x[i, ]` belongs to cluster `j`.

**Note**

`get.idr` returns a vector where the  $i$ 'th entry is the posterior probability that observation  $i$  is irreproducible. It is a simple wrapper for `get.prob`. From **GMCM** version 1.1 it is an internal. Use `get.prop` or `get.IDR` instead. However, the function can still be accessed with `GMCM::get.idr`.

**Author(s)**

Anders Ellern Bilgrau ([abilgrau@math.aau.dk](mailto:abilgrau@math.aau.dk))

**References**

- Li, Q., Brown, J. B. J. B., Huang, H., & Bickel, P. J. (2011). Measuring reproducibility of high-throughput experiments. *The Annals of Applied Statistics*, *5*(3), 1752-1779. doi:10.1214/11-AOAS466
- Tewari, A., Giering, M., & Raghunathan, A. (2011). Parametric Characterization of Multimodal Distributions with Non-gaussian Modes. *IEEE 11th International Conference on Data Mining Workshops, 2011*, 286-292. doi:10.1109/ICDMW.2011.135

**Examples**

```
set.seed(1123)

# True parameters
true.par <- c(0.9, 2, 0.7, 0.6)

# Simulation of data from the GMCM model
data <- SimulateGMCMData(n = 1000, par = true.par, d = 2)

# Initial parameters
init.par <- c(0.5, 1, 0.5, 0.9)

# Nelder-Mead optimization
nm.par <- fit.meta.GMCM(data$u, init.par = init.par, method = "NM")

# Get IDR values
res <- get.IDR(data$u, nm.par, threshold = 0.05)

# Plot results
plot(data$u, col = res$Khat, pch = c(3,16)[data$K])
```

---

`is.theta`*Check if parameters are valid*

---

**Description**

Function to check whether the argument is coherent and in the correct format.

**Usage**

```
is.theta(theta)
```

**Arguments**

`theta` A list on the theta-form described in [rtheta](#)

**Value**

logical. Returns TRUE if theta is coherent and in the correct format. Otherwise, the function returns FALSE with an accompanying warning message of the problem.

**Author(s)**

Anders Ellern Bilgrau (abilgrau@math.aau.dk)

**See Also**

[rtheta](#)

**Examples**

```
theta1 <- rtheta() # Create a random correctly formatted theta
is.theta(theta1)

theta2 <- rtheta(d = 3, m = 5)
theta2$m <- 6 # m is now incoherent with the number of components
is.theta(theta2)

theta3 <- rtheta(d = 4, m = 2)
theta3$sigma$comp1[1, 2] <- 0 # Making the covariance matrix non-symmetric
is.theta(theta3)

theta4 <- rtheta(d = 10, m = 10)
theta4$sigma$comp1[1, 1] <- 0 # Destroy positive semi-definiteness
is.theta(theta4)
```

---

rtheta

*Get random parameters for the Gaussian mixture (copula) model*


---

**Description**

The function generates a random set parameters for the Gaussian mixture model (GMM) and Gaussian mixture copula model (GMCM).

**Usage**

```
rtheta(m = 3, d = 2)
```

**Arguments**

m	The number of components in the mixture.
d	The dimension of the mixture distribution.

**Value**

A named list of parameters with the 4 elements:

m	An integer giving the number of components in the mixture. Default is 3.
d	An integer giving the dimension of the mixture distribution. Default is 2.
pie	A numeric vector of length m of mixture proportions between 0 and 1 which sums to one.
mu	A list of length m of numeric vectors of length d for each component.
sigma	A list of length m of variance-covariance matrices (of size d times d) for each component.

**Note**

The function `is.theta` checks whether or not theta is in the correct format.



**Author(s)**

Anders Ellern Bilgrau (abilgrau@math.aau.dk)

**See Also**

[is.theta](#)

**Examples**

```
rtheta()
rtheta(d = 5, m = 2)

test <- rtheta()
is.theta(test)
```

---

SimulateGMCMData

*Simulation from the Gaussian mixture (copula) model*

---

**Description**

Functions to simulate data from the Gaussian mixture model (GMM) and the Gaussian mixture copula model (GMCM).

**Usage**

```
SimulateGMCMData(n = 1000, par, d = 2, theta, ...)
```

```
SimulateGMMData(n = 1000, theta = rtheta(...), ...)
```

**Arguments**

n	Integer. The number of realizations drawn from the model. Default is 1000.
par	A vector of parameters of length 4 where par[1] is the mixture proportion, par[2] is the mean, par[3] is the standard deviation, and par[4] is the correlation.
d	The number of dimensions (or, equivalently, experiments) in the mixture distribution.
theta	A list of parameters for the model as described in <a href="#">rtheta</a> .
...	In <code>SimulateGMCMData</code> the arguments are passed to <code>SimulateGMMData</code> . In <code>SimulateGMMData</code> the arguments are passed to <a href="#">rtheta</a> .

**Details**

The `par` argument specifies the parameters of the Li et. al. (2011) GMCM. The `theta` argument specifies an arbitrary GMCM. Either one can be supplied. If both are missing, random parameters are chosen for the general model.

**Value**

SimulateGMCMData returns a list of length 4 with elements:

u	A matrix of the realized values of the GMCM.
z	A matrix of the latent GMM realizations.
K	An integer vector denoting the component from which the realization comes.
theta	A list containing the used parameters for the simulations with the format described in <a href="#">rtheta</a> .

SimulateGMMData returns a list of length 3 with elements:

z	A matrix of GMM realizations.
K	An integer vector denoting the component from which the realization comes.
theta	As above and in <a href="#">rtheta</a> .

**Author(s)**

Anders Ellern Bilgrau ([abilgrau@math.aau.dk](mailto:abilgrau@math.aau.dk))

**See Also**

[rtheta](#)

**Examples**

```
set.seed(2)

# Simulation from the GMM
gmm.data1 <- SimulateGMMData(n = 200, m = 3, d = 2)
str(gmm.data1)

# Plotting the simulated data
plot(gmm.data1$z, col = gmm.data1$K)

# Simulation from the GMCM
gmcm.data1 <- SimulateGMCMData(n = 1000, m = 4, d = 2)
str(gmcm.data1)

# Plot the 2nd simulation
par(mfrow = c(1,2))
plot(gmcm.data1$z, col = gmcm.data1$K)
plot(gmcm.data1$u, col = gmcm.data1$K)

# Simulation from the special case of GMCM
theta <- meta2full(c(0.7, 2, 1, 0.7), d = 3)
gmcm.data2 <- SimulateGMCMData(n = 5000, theta = theta)
str(gmcm.data2)

# Plotting the 3rd simulation
par(mfrow=c(1,2))
```

```
plot(gmcm.data2$z, col = gmcm.data2$K)  
plot(gmcm.data2$u, col = gmcm.data2$K)
```

---

u133VsExon

*Reproducibility between U133 plus 2 and Exon microarrays*

---

## Description

This dataset contains a data.frame of unadjusted P-values for differential expression between germinal center cells and other B-cells within tonsils for two different experiments. The experiments differ primarily in the microarray platform used. The first column corresponds to the evidence from the Affymetrix GeneChip Human Genome U133 Plus 2.0 Array. The second column corresponds to the Affymetrix GeneChip Human Exon 1.0 ST Array.

## Format

The format of the data.frame is:

```
'data.frame': 19577 obs. of 2 variables:  
 $ u133: num  0.17561 0.00178 0.005371 0.000669 0.655261 ...  
 $ exon: num  1.07e-01 6.74e-10 1.51e-03 6.76e-05 3.36e-01 ...
```

## Details

Further details can be found in Bergkvist et al. (2014) and Rasmussen and Bilgrau et al. (2014).

## Author(s)

Anders Ellern Bilgrau <abilgrau (at) math.aau.dk>

## References

Bergkvist, Kim Steve, Mette Nyegaard, Martin Boegsted, Alexander Schmitz, Julie Stoeve Boedker, Simon Mylius Rasmussen, Martin Perez-Andres et al. "Validation and implementation of a method for microarray gene expression profiling of minor B-cell subpopulations in man." BMC immunology 15, no. 1 (2014): 3.

Rasmussen SM, Bilgrau AE, Schmitz A, Falgreen S, Bergkvist KS, Tramm AM, Baech J, Jacobsen CL, Gaihede M, Kjeldsen MK, Boedker JS, Dybkaer K, Boegsted M, Johnsen HE (2014). "Stable Phenotype Of B-Cell Subsets Following Cryopreservation and Thawing of Normal Human Lymphocytes Stored in a Tissue Biobank." Cytometry Part B: Clinical Cytometry. ISSN 1552-4957.

**Examples**

```
data(u133VsExon)
str(u133VsExon)

# Plot P-values
plot(u133VsExon, cex = 0.5)

# Plot ranked and scaled P-values
plot(Uhat(1-u133VsExon), cex = 0.5)
```

---

Uhat

*Fast ranking function*

---

**Description**

Function for computing the scaled ranks for each column of the input matrix. In other words, the values are ranked column-wise and divided by  $nrow(x) + 1$ . A "1334" ranking scheme is used where the lowest values are awarded rank 1, second lowest value rank 2, and ties are given the maximum available rank.

**Usage**

```
Uhat(x)
```

**Arguments**

**x** A numeric matrix of observations to be ranked. Rows correspond to features and columns to experiments.

**Value**

A matrix with the same dimensions as **x** of the scaled ranks.

**Author(s)**

Anders Ellern Bilgrau (abilgrau@math.aau.dk)

**See Also**

[SimulateGMMData](#), [SimulateGMCMDData](#)

**Examples**

```
data <- SimulateGMMData()
par(mfrow = c(1,2))
plot(data$z, xlab = expression(z[1]), ylab = expression(z[2]))
plot(Uhat(data$z),
     xlab = expression(hat(u)[1]),
     ylab = expression(hat(u)[2]))
```

# Index

- \*Topic **datasets**,
  - freshVsFrozen, 11
  - u133VsExon, 19
- \*Topic **data**
  - freshVsFrozen, 11
  - u133VsExon, 19
- choose.theta, 3, 3, 8
- dmvnormal, 5
- EMAlgorithm, 6
- fit.full.GMCM, 2, 3, 7
- fit.full.gmcm (fit.full.GMCM), 7
- fit.meta.GMCM, 2, 3, 9
- fit.meta.gmcm (fit.meta.GMCM), 9
- freshVsFrozen, 11
- full2meta, 3, 12
- get.IDR, 2, 3, 13
- get.idr (get.IDR), 13
- get.prob, 2, 3, 8
- get.prob (get.IDR), 13
- GMCM (GMCM-package), 2
- GMCM-package, 2
- idr, 3
- is.theta, 15, 16, 17
- kmeans, 4
- meta2full, 3
- meta2full (full2meta), 12
- optim, 7–10
- PseudoEMAlgorithm, 7, 10
- qgmm.marginal, 14
- rmvnormal (dmvnormal), 5
- rtheta, 3, 4, 6–8, 12–15, 16, 17, 18
- SimulateGMCMData, 2, 3, 17, 20
- SimulateGMMData, 3, 20
- SimulateGMMData (SimulateGMCMData), 17
- u133VsExon, 19
- Uhat, 3, 20